STATA LINEARIZED DYNAMIC STOCHASTIC GENERAL EQUILIBRIUM REFERENCE MANUAL

RELEASE 15

A Stata Press Publication
StataCorp LLC
College Station, Texas

Version 15

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845
Typeset in TEX

ISBN-10: 1-59718-259-1
ISBN-13: 978-1-59718-259-1

The suggested citation for this software is

StataCorp. 2017. *Stata: Release 15*. Statistical Software. College Station, TX: StataCorp LLC.

Contents

Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals. For example,

[U] 26 Overview of Stata estimation commands
[XT] **xtabond**
[D] **reshape**

The first example is a reference to chapter 26, *Overview of Stata estimation commands*, in the *User's Guide*; the second is a reference to the `xtabond` entry in the *Longitudinal-Data/Panel-Data Reference Manual*; and the third is a reference to the `reshape` entry in the *Data Management Reference Manual*.

All the manuals in the Stata Documentation have a shorthand notation:

[GSM]	*Getting Started with Stata for Mac*
[GSU]	*Getting Started with Stata for Unix*
[GSW]	*Getting Started with Stata for Windows*
[U]	*Stata User's Guide*
[R]	*Stata Base Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[D]	*Stata Data Management Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[DSGE]	*Stata Linearized Dynamic Stochastic General Equilibrium Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[TE]	*Stata Treatment-Effects Reference Manual: Potential Outcomes/Counterfactual Outcomes*
[I]	*Stata Glossary and Index*
[M]	*Mata Reference Manual*

Title

intro — Introduction

Description Remarks and examples Also see

Description

DSGE stands for dynamic stochastic general equilibrium. DSGE models are multivariate time-series models that are used in economics, in particular, macroeconomics, for policy analysis and forecasting. These models are systems of equations that are typically derived from economic theory. As such, the parameters are often directly interpretable based on theory. DSGE models are unique in that equations in the system allow current values of variables to depend not only on past values but also on expectations of future values.

The `dsge` command estimates parameters of linearized DSGE models.

Remarks and examples

We recommend that you read this manual beginning with [DSGE] **intro 1** and then continue with the remaining introductions. In these introductions, we will introduce DSGE models, show you how to use the `dsge` command, walk you through worked examples of classic models, and present solutions to common stumbling blocks.

[DSGE] **intro 1** and [DSGE] **intro 2** are essential reading. Read them first. Here you will find an overview of DSGE models, descriptions of concepts used throughout the manual, discussion of assumptions, a first worked example, and an introduction to the syntax.

[DSGE] **intro 1** Introduction to DSGE models
[DSGE] **intro 2** Learning the syntax

[DSGE] **intro 3** focuses on classical DSGE models. It includes a series of examples that illustrate model solution, model estimation, and postestimation procedures for simple variants of common models.

[DSGE] **intro 3** Classic DSGE examples
[DSGE] **intro 3a** New Keynesian model
[DSGE] **intro 3b** New Classical model
[DSGE] **intro 3c** Financial frictions model

[DSGE] **intro 4** discusses some common problems and solutions for them. The structural equations of the linearized DSGE model must have a specific structure so that the model can be solved. Often DSGE models are written using intuitive forms that do not have this structure. These intuitive forms can be rewritten in a logically equivalent form that has the structure required for solution. [DSGE] **intro 4** provides an overview of this topic and examples demonstrating solutions.

[DSGE] **intro 4** Writing a DSGE in a solvable form
[DSGE] **intro 4a** Specifying a shock on a control variable
[DSGE] **intro 4b** Including a lag of a control variable
[DSGE] **intro 4c** Including a lag of a state variable
[DSGE] **intro 4d** Including an expectation of a control dated by more than one period ahead
[DSGE] **intro 4e** Including a second-order lag of a control
[DSGE] **intro 4f** Including an observed exogenous variable
[DSGE] **intro 4g** Correlated state variables

[DSGE] **intro 5**–[DSGE] **intro 8** discuss technical issues. These introductions are essential reading, even though they are last.

[DSGE] **intro 5** Stability conditions
[DSGE] **intro 6** Identification
[DSGE] **intro 7** Convergence problems
[DSGE] **intro 8** Wald tests vary with nonlinear transforms

The main command entries are references for syntax and implementation details. All the examples are in the introductions discussed above.

[DSGE] **dsge** Linearized dynamic stochastic general equilibrium models
[DSGE] **dsge postestimation** Postestimation tools for dsge
[DSGE] **estat policy** Display policy matrix
[DSGE] **estat stable** Display stability results
[DSGE] **estat transition** Display state transition matrix

Also see

[DSGE] **intro 1** — Introduction to DSGEs and dsge

Title

intro 1 — Introduction to DSGEs and dsge

Description Remarks and examples References Also see

Description

In this entry, we introduce DSGE models and the `dsge` command. We begin with an overview of DSGE models. We then illustrate the complete process of DSGE modeling by doing an example from start to finish. In this example, we demonstrate how to describe a model in its original nonlinear form, write it in a corresponding linearized form, estimate the parameters of the linearized model, and interpret the results. We conclude by showing how this example fits into the general DSGE modeling framework.

Remarks and examples

Remarks are presented under the following headings:

Introduction to DSGE models

DSGE models are models for multiple time series used in macroeconomics and finance. These models are systems of equations that are motivated by economic theory and in which expectations of future values of variables play an important role. Because these models come from theory, the parameters of these models can typically be directly interpreted in terms of the motivating theory. DSGE models are used for macroeconomic policy analysis and forecasting.

In DSGE models, individuals' actions are summarized by decision rules that take the form of nonlinear systems of dynamic equations. These decision rules often come from dynamic stochastic optimization problems. A DSGE model consists of these decision rules, plus any aggregation conditions, resource or budget constraints, and stochastic processes for exogenous variables.

Because the model's equations are the solution to dynamic optimization problems, model equations can feature expectations of future variables. These expectations are endogenous. In DSGE models, expectations of future variables correspond to their conditional mean as implied by the model. In other words, individuals' expectations of future values are correct, on average. Such expectations are said to be model-consistent expectations or rational expectations.

There are three kinds of variables in DSGE models: control variables, state variables, and shocks. The terminology is taken from the state-space and optimal control literatures. In DSGE models, the concepts of exogeneity and endogeneity are understood relative to a time period. A state variable is fixed, or exogenous, in a given time period. The system of equations then determines the value of the state variable one period in the future. On the other hand, the system of equations determines the value of a control variable in the current time period. State variables evolve over time and may depend on control variables. State variables can also be correlated. Control variables in a DSGE model can be either observed or unobserved. State variables are always unobserved.

DSGE models can be written in multiple forms. The model based on economic theory may consist of equations that are nonlinear both in the variables and in the parameters. A DSGE model is said to be linear, or linearized, when the model equations are linear in the variables. DSGE models are commonly linearized prior to analysis. After linearization, the model variables are expressed as deviations from a steady state.

The DSGE model must be solved prior to estimation. Dynamic linear models are easier to solve and fit than dynamic nonlinear models, particularly when future expectations are incorporated into the model.

In any analysis of simultaneous equations systems, to solve a model means to write the model's endogenous variables as functions of its exogenous variables. In DSGE models, the analogous solution concept is to write the model's control variables in terms of its state variables. The model's solution consists of a system of equations relating the control variables to the state variables and a system of equations describing the evolution of the state variables over time. The solution to a DSGE model thus takes the form of a state-space model. The solution to a DSGE model is a crucial object for both estimation and analyses after estimation. Both the likelihood function and the impulse–response functions are formed from the model solution.

`dsge` solves and estimates the parameters of linearized DSGE models.

General introductions to DSGE modeling are available in Ljungqvist and Sargent (2012) and Woodford (2003). Canova (2007) and DeJong and Dave (2011) describe parameter estimation using DSGE models.

How to write down a DSGE

Writing down a DSGE is a two-step process. First, we write down (potentially) nonlinear structural equations that come from theory, usually economic theory. In many such theories, individuals do the best that they can given their constraints. This idea is formalized in dynamic stochastic optimization problems. The solutions to these problems are the nonlinear form of DSGE models. Second, we write down the linearized model corresponding to the nonlinear structural model.

Consider the following nonlinear model, similar to that in Woodford (2003, chap. 4). The model consists of equations that describe the behavior of households, firms, and a central bank. Interactions among these actors produce a model of inflation, output growth, and the interest rate. Models of this type are popular in academic and policy settings and are used to describe and analyze monetary policy.

Household optimization generates an equation that relates current output Y_t to the expected value of a function of tomorrow's output Y_{t+1}, tomorrow's inflation Π_{t+1}, and the current nominal interest rate R_t,

$$\frac{1}{Y_t} = \beta E_t\left[\frac{1}{Y_{t+1}}\frac{R_t}{\Pi_{t+1}}\right] \tag{1a}$$

where β is a parameter that captures households' willingness to delay consumption.

Optimization by firms generates an equation that links the current deviation of inflation from its steady state, $\Pi_t - \Pi$, to the expected value of the deviation of inflation from its steady state in the future, $E_t\left(\Pi_{t+1} - \Pi\right)$, and to the ratio of actual output, Y_t, to the natural level of output, Z_t,

$$\phi\left(\Pi_t - \Pi\right) + \theta - 1 = \theta\left(\frac{Y_t}{Z_t}\right)^{\eta-1} + \beta\phi E_t\left(\Pi_{t+1} - \Pi\right) \tag{2a}$$

where ϕ, η, and θ are parameters linked to the pricing decision of firms. Firms are not affected by inflation per se; they are affected only by deviations of inflation from its steady-state value.

Finally, there is an equation that describes central bank policy. The central bank adjusts the interest rate in response to inflation and, perhaps, to other factors that we leave unmodeled. The equation for the central bank policy is

$$\frac{R_t}{R} = \left(\frac{\Pi_t}{\Pi}\right)^{1/\beta} e^{u_t} \tag{3a}$$

where R is the steady-state value of the interest rate and u_t is a state variable that captures all movements in the interest rate that are not driven by inflation.

Using the dsge command

Before you can use `dsge`, you must make sure certain things about your data and your model are true.

First, you must linearize the model by using deviations from steady state. `dsge` implements a check for linearity and will complain if it detects nonlinearities. We do not discuss how to linearize the equations in this manual. See, for example, DeJong and Dave (2011, chap. 2), which provides an overview of the linearization process.

Second, the data must be `tsset` prior to using `dsge`. Using `tsset` on the data allows us to use time-series operators.

Finally, the series in your model must have zero mean and be weakly stationary. `dsge` will demean the series for you, but you need to ensure that your data are weakly stationary before specifying them in the `dsge` command.

Writing down linearized DSGEs

The model in (1a)–(3a) is nonlinear. We now need to write the model in its corresponding linearized form, which we show below. Throughout this manual, we use lowercase letters to denote percentage deviations of variables from the steady state. The linearized versions of the above equations are

$$y_t = E_t y_{t+1} - (r_t - E_t\pi_{t+1}) \tag{1b}$$

$$\pi_t = \beta E_t\pi_{t+1} + \kappa(y_t - z_t) \tag{2b}$$

$$r_t = \frac{1}{\beta}\pi_t + u_t \tag{3b}$$

The new parameter κ is a complicated function of the underlying parameters from (2a). Those underlying parameters cannot be separately identified in this model, but κ can be.

We have implicitly constrained some of the coefficients; for example, the coefficient on the interest rate in the output equation is constrained to -1.

Note that we have not yet specified stochastic processes for z_t and u_t.

Data preparation

We have data on price levels and interest rates in `rates.dta`. These data were obtained from the Federal Reserve Economic Database (FRED), which contains many macroeconomic and financial time series; see [D] **import fred**.

```
. use http://www.stata-press.com/data/r15/rates2
(Federal Reserve Economic Data - St. Louis Fed, 2017-02-10)
. describe
Contains data from http://www.stata-press.com/data/r15/rates2.dta
  obs:           281                          Federal Reserve Economic Data -
                                                St. Louis Fed, 2017-02-10
 vars:             5                          26 Apr 2017 21:22
 size:         6,182
-------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
-------------------------------------------------------------------------------
datestr         str10   %-10s                 Observation date
daten           int     %td                   Numeric (daily) date
gdpdef          float   %9.0g                 GDP deflator GDPDEF
r               float   %9.0g                 Federal funds rate FEDFUNDS
dateq           int     %tq                   Quarterly date
-------------------------------------------------------------------------------
Sorted by: dateq
```

The dataset includes the price level and the interest rate. But the model is written in terms of the inflation rate. For quarterly data, the inflation rate is conventionally obtained as 400 times the difference in log of price variable. Therefore, we begin by generating an inflation rate variable `p` by using the `L.` lag operator.

```
. generate p = 400*(ln(gdpdef) - ln(L.gdpdef))
(2 missing values generated)
. label variable p "Inflation rate"
```

We now have inflation and interest rates. They are not mean zero, but `dsge` will demean the data for us.

Specifying the model to dsge

We make one final modification to the model equations before estimating the parameters. Woodford (2003) rewrites the model in (1b)–(3b) by defining $x_t = y_t - z_t$ as the output gap. We do the same. Substituting in x_t gives us the three-equation system

$$x_t = E_t x_{t+1} - (r_t - E_t\pi_{t+1} - g_t) \tag{4}$$

$$\pi_t = \beta E_t \pi_{t+1} + \kappa x_t \tag{5}$$

$$r_t = \frac{1}{\beta}\pi_t + u_t \tag{6}$$

where $g_t = E_t(z_{t+1}) - z_t$ is a state variable. Equations (4)–(6) are the structural form of how the endogenous control variables x_t, π_t, and r_t evolve as functions of the exogenous state variables g_t and u_t.

We complete the model by specifying processes for how the state variables evolve. Per standard practice, both are modeled as first-order autoregressive processes.

$$u_{t+1} = \rho_u u_t + \epsilon_{t+1} \tag{7}$$
$$g_{t+1} = \rho_g g_t + \xi_{t+1} \tag{8}$$

The variables ξ_{t+1} and ϵ_{t+1} are shocks to the state variables.

Equations (4)–(6) are the linearized DSGE model derived from (1a)–(3a). Together with (7)–(8) for the evolution of state variables, this is a complete model whose parameters can be estimated.

The command to estimate the parameters in the system of (4)–(8) is

```
. dsge (p   = {beta}*E(F.p) + {kappa}*x)            ///
       (x   = E(F.x) -(r - E(F.p) - g), unobserved) ///
       (r   = (1/{beta})*p + u)                     ///
       (F.u = {rhou}*u, state)                      ///
       (F.g = {rhog}*g, state)
```

Each equation is bound in parentheses. The equations look almost identical to the system in (4)–(8). Because a model has as many variables as it has equations, each variable will appears on the left-hand side of one and only one equation.

The equation options `unobserved` and `state` modify how `dsge` interprets the equations we supply. Equations may be specified in any order; in this manual, we typically write control equations and then state equations, but you don't have to follow that convention.

`p`, `x`, and `r` are our control variables p_t, x_t, and r_t. Each appears on the left-hand side of one equation and can appear on the right-hand side of as many equations as we like. There are two state equations with shocks. The model must have the same number of shocks as observed control variables. Therefore, we can treat only two of three control variables as observed. Of our three control variables, the output gap is the most plausible to be unobserved. Thus we model the output gap as unobserved, and we model the inflation rate and the interest rate as observed. To specify inflation and the interest rate as observed, we only need to write their equations. We specify the output gap as unobserved using the `unobserved` option.

`u` and `g` are the state variables u_t and g_t. Recall that state variables are fixed in the current period, so we specify how they evolve through time by modeling the one-period lead—hence, the `F.` on the left-hand side of each state equation. The state equations specify how the state variable evolves as a function of the current state variables and, possibly, the control variables.

The shocks ϵ_t and ξ_t enter the system through the state equations of their corresponding variable. By default, a shock is attached to each state equation. So, when we typed

```
. dsge ... (F.u = {rhou}*u, state) ...
```

the underlying equation is what we wrote in (7). If a state variable is treated as deterministic in your model, then it will not have a shock. For example, capital accumulation is often treated as deterministic. To include an equation for a state variable without a shock, we would include the `noshock` option within the equation.

Expectations of future variables appear within the `E()` operator and use Stata's `F.` operator. For example, type `E(F.x)` to specify $E_t(x_{t+1})$.

The parameters we want to estimate are bound in braces.

For more details on the `dsge` syntax, see [DSGE] **intro 2**.

Parameter estimation and interpretation

We estimate the parameters of the model in (4)–(8). These equations are much discussed in the monetary economics literature. Equation (4) is known as the output-gap Euler equation. Equation (5) is known as a New Keynesian Phillips Curve, and the parameter κ is known as the slope of the Phillips curve. In New Keynesian models, prices depend on output, and κ is a measure of that dependence. Equation (6) is known as a Taylor rule, after Taylor (1993). The coefficient on inflation in a Taylor rule is a commonly discussed parameter. β has two roles in the model above. It relates current inflation deviations to expected future inflation deviations, and it relates interest rate deviations to inflation deviations.

```
. dsge (p = {beta}*E(F.p) + {kappa}*x)
>      (x = E(F.x) -(r - E(F.p) - g), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.g = {rhog}*g, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -13931.564
Iteration 1:   log likelihood = -1301.5118  (backed up)
Iteration 2:   log likelihood = -1039.6984  (backed up)
Iteration 3:   log likelihood = -905.70867  (backed up)
Iteration 4:   log likelihood = -842.76867  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -812.04209  (backed up)
Iteration 6:   log likelihood = -786.76609
Iteration 7:   log likelihood = -777.19779
Iteration 8:   log likelihood =  -768.8383
Iteration 9:   log likelihood =  -768.1368
Iteration 10:  log likelihood = -768.09519
Iteration 11:  log likelihood = -768.09383
Iteration 12:  log likelihood = -768.09383
DSGE model
Sample: 1954q3 - 2016q4                         Number of obs     =        250
Log likelihood = -768.09383
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5112881	.075791	6.75	0.000	.3627404	.6598359
kappa	.1696296	.0475492	3.57	0.000	.076435	.2628243
rhou	.6989189	.0449192	15.56	0.000	.6108789	.7869588
rhog	.9556407	.0181342	52.70	0.000	.9200983	.9911831
sd(e.u)	2.317589	.2988024			1.731947	2.90323
sd(e.g)	.6147348	.0973279			.4239757	.8054939

Two of the parameters have structural interpretations. The parameter `kappa` is the slope of the Phillips curve. Theory predicts that this parameter will be positive, and indeed our estimate is positive.

The parameter `beta` is the inverse of the coefficient on inflation in the interest rate equation. We can obtain an estimate of $1/\beta$, which is interpreted as the degree to which the central bank responds to movements in inflation, by using `nlcom`.

```
. nlcom 1/_b[beta]
       _nl_1:  1/_b[beta]
```

	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
_nl_1	1.955844	.2899255	6.75	0.000	1.387601	2.524088

A typical value for $1/\beta$ in the literature is 1.5. Our point estimate is around 2.

Postestimation

Policy and transition matrices

The matrix of parameters in the state-space form that specifies how the state variables affect the control variables is known as the policy matrix. Each policy matrix parameter is the effect of a one-unit shock to a state variable on a control variable.

▷ Example 1: Obtaining the policy matrix

We use `estat policy` to view these results.

```
. estat policy
Policy matrix
```

		Coef.	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
p							
	u	-.4170859	.0389324	-10.71	0.000	-.4933919	-.3407799
	g	.881884	.2330573	3.78	0.000	.4251001	1.338668
x							
	u	-1.580153	.3926336	-4.02	0.000	-2.3497	-.8106049
	g	2.658667	.9045286	2.94	0.003	.885823	4.43151
r							
	u	.1842449	.056798	3.24	0.001	.072923	.2955669
	g	1.724828	.2210259	7.80	0.000	1.291625	2.158031

Results are listed equation by equation. The first block is the policy equation for inflation `p` and writes it as a function of the state variables alone. A unit shock to the state `u` reduces inflation by an estimated 0.417, and a unit shock to `g` raises inflation by an estimated 0.882.

◁

The matrix of parameters that specifies the dynamic process for the state variables is known as the state transition matrix. The state transition equation relates the future values of the state variables to their values in the current period. Each state transition matrix parameter is the effect of a one-unit shock to a state variable on its one-period-ahead mean.

▷ Example 2: Obtaining the transition matrix

```
. estat transition
Transition matrix of state variables
```

		Coef.	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
F.u							
	u	.6989189	.0449192	15.56	0.000	.6108789	.7869588
	g	1.11e-16	7.11e-12	0.00	1.000	-1.39e-11	1.39e-11
F.g							
	u	0	(omitted)				
	g	.9556407	.0181342	52.70	0.000	.9200983	.9911831

Both state variables are modeled as autoregressive processes, so the results in `estat transition` repeat the estimates of `rhou` and `rhog` from the `dsge` output. In this case, the other entries in the state transition matrix are 0 or differ from 0 only because of lack of numerical precision. In more complicated models, such as those in which a state equation depends on a control variable, the state transition matrix will contain new information about that state variable.

◁

Impulse responses

The state-space form allows us to trace the path of a control or state in response to a shock to a state. This path is known as an impulse–response function (IRF). `irf` after `dsge` estimates IRFs, and it puts the named set of estimates into an `.irf` file, whose results can be displayed using `irf graph` or `irf table`.

▷ Example 3: Graphing an IRF

To graph the IRF, we first create the `nkirf.irf` file and set it as the active `.irf` file using the `irf set` command.

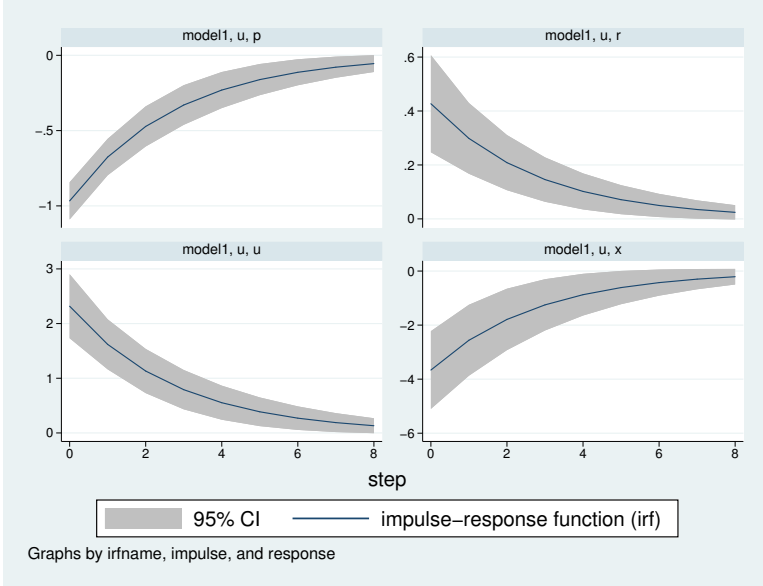
```
. irf set nkirf.irf
(file nkirf.irf created)
(file nkirf.irf now active)
```

Next, we use `irf create` to estimate a complete set of impulse responses based on our `dsge` command. A complete set of impulse responses is an impulse to each shock and the response of each state and control variable to that impulse. For the model in this example, `irf create` generates an impulse to `e.u` and `e.g`, then stores the response to that impulse on `p`, `x`, `r`, `g`, and `u`. The results are stored in the `nkirf.irf` file.

```
. irf create model1
(file nkirf.irf updated)
```

We now use `irf graph` to plot the impulse responses. The `impulse()` and `response()` options control which impulse and which responses are chosen. To view the response of `p`, `x`, `r`, and `u` to a shock to `u`, we type

```
. irf graph irf, impulse(u) response(x p r u) byopts(yrescale)
```



The state variable `u` models movements in the interest rate that occur for reasons other than the feedback between inflation and the interest rate. A shock to `u` is effectively a surprise increase in the interest rate, and the IRF traces out how this shock causes temporary decreases to inflation (top-left graph) and to the output gap (bottom-right graph).

◁

Forecasts

The `forecast` suite of commands produces dynamic forecasts from the fitted model.

▷ Example 4: Out-of-sample forecast

We first store the `dsge` estimation results.

```
. estimates store dsge_est
```

We use `tsappend()` to extend the dataset by 3 years, or 12 quarters.

```
. tsappend, add(12)
```

To set up a forecast, we perform three steps. `forecast create` initializes a new forecasting model, which we name `dsgemodel`.

```
. forecast create dsgemodel
  Forecast model dsgemodel started.
```

Next, we add the estimates from `dsge` to the forecasting model using `forecast estimates`.

```
. forecast estimates dsge_est
  Added estimation results from dsge.
  Forecast model dsgemodel now contains 2 endogenous variables.
```

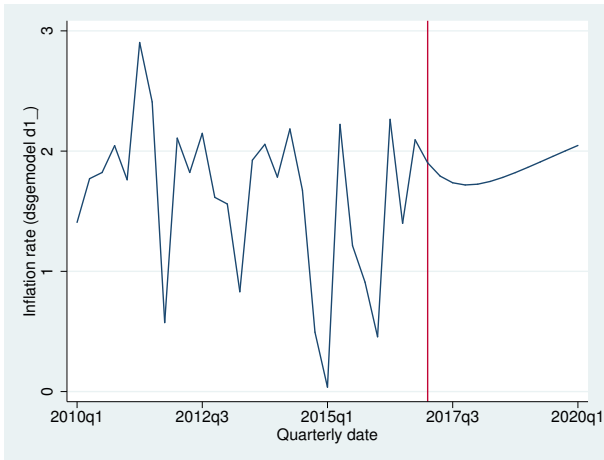
This command adds the estimates stored in `dsge_est` to the model `dsgemodel`. We now produce dynamic forecasts beginning in the first quarter of 2017 using `forecast solve`. The `prefix(d1_)` option specifies the `d1_` prefix that will be given to the variables created by `forecast`. We also request that dynamic forecasts begin in the first quarter of 2017 with the `begin(tq(2017q1))` option.

```
. forecast solve, prefix(d1_) begin(tq(2017q1))
Computing dynamic forecasts for model dsgemodel.
------------------------------------------------------------
Starting period:  2017q1
Ending period:    2020q1
Forecast prefix:  d1_

2017q1:  .............
2017q2:  .............
2017q3:  .............
2017q4:  .............
2018q1:  .............
2018q2:  .............
2018q3:  .............
2018q4:  .............
2019q1:  .............
2019q2:  .............
2019q3:  .............
2019q4:  .............
2020q1:  .............

Forecast 2 variables spanning 13 periods.
------------------------------------------------------------
```

The dynamic forecast begins in the first quarter of 2017, so all forecasts are out of sample.

We can graph the forecast for inflation `d1_p` using `tsline`.

```
. tsline d1_p if tin(2010q1, 2021q1), tline(2017q1)
```



The model forecasts that inflation will smoothly return to its long-run value, the sample mean. ◁

We can also begin the forecast during a time period for which observations are available.

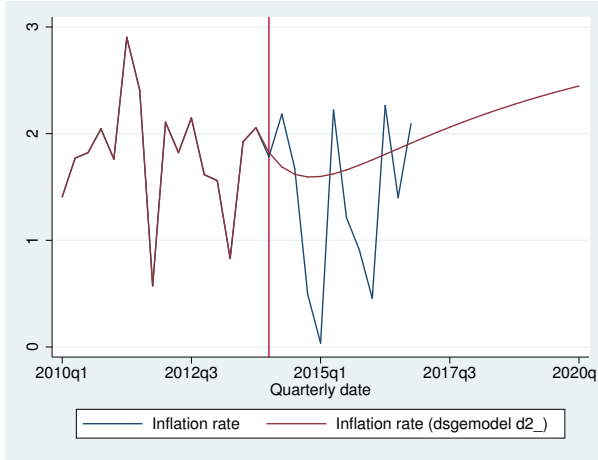
▷ Example 5: Within-sample forecast

Specifying the `begin(tq(2014q1))` option produces dynamic forecasts beginning in the first quarter of 2014, so we can compare the forecast for 2014–2016 with the actual observations over that period.

```
. forecast solve, prefix(d2_) begin(tq(2014q1))
Computing dynamic forecasts for model dsgemodel.
------------------------------------------------------------

Starting period:  2014q1
Ending period:    2020q1
Forecast prefix:  d2_

2014q1:  ..............
2014q2:  ..............
2014q3:  ..............
2014q4:  ..............
2015q1:  ..............
2015q2:  ..............
2015q3:  ..............
2015q4:  ............
2016q1:  ............
2016q2:  ............
2016q3:  ............
2016q4:  ............
2017q1:  ............
2017q2:  ............
2017q3:  ............
2017q4:  ............
2018q1:  ............
2018q2:  ............
2018q3:  ............
2018q4:  ............
2019q1:  ............
2019q2:  ............
2019q3:  ............
2019q4:  ............
2020q1:  ............

Forecast 2 variables spanning 25 periods.
------------------------------------------------------------
```

```
. tsline p d2_p if tin(2010q1, 2021q1), tline(2014q1)
```

We plot both the observed inflation and the forecast.



The forecast captures the general upward trend in inflation from 2014–2016, but it does not predict the variation in inflation around the upward trend.

◁

Structural and reduced forms of DSGE models

Now that we have worked an example, we show how it fits in the more general formulation of DSGE models. The model in (4)–(8) is an example of a linearized DSGE model. In general, a linearized DSGE model can be expressed as

$$\mathbf{A}_0\mathbf{y}_t = \mathbf{A}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{A}_2\mathbf{y}_t + \mathbf{A}_3\mathbf{x}_t \tag{9}$$

$$\mathbf{B}_0\mathbf{x}_{t+1} = \mathbf{B}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{B}_2\mathbf{y}_t + \mathbf{B}_3\mathbf{x}_t + \mathbf{C}\boldsymbol{\epsilon}_{t+1} \tag{10}$$

where $\mathbf{y}_t$ is a vector of control variables, $\mathbf{x}_t$ is a vector of state variables, and $\boldsymbol{\epsilon}_t$ is a vector of shocks. $\mathbf{A}_0$ through $\mathbf{A}_3$ and $\mathbf{B}_0$ through $\mathbf{B}_3$ are matrices of parameters. We require that $\mathbf{A}_0$ and $\mathbf{B}_0$ be diagonal matrices. The entries in $\mathbf{A}_i$ and $\mathbf{B}_j$ are all functions of the structural parameters, which we denote by vector $\boldsymbol{\theta}$. Economic theory places restrictions on the $\mathbf{A}_i$ and $\mathbf{B}_j$ matrices. $\mathbf{C}$ is a selection matrix that determines which state variables are subject to shocks.

The state-space form of the model is given by

$$\mathbf{y}_t = \mathbf{G}\mathbf{x}_t \tag{11}$$

$$\mathbf{x}_{t+1} = \mathbf{H}\mathbf{x}_t + \mathbf{M}\boldsymbol{\epsilon}_{t+1} \tag{12}$$

where $\mathbf{y}_t$ is a vector of control variables, $\mathbf{x}_t$ is a vector of state variables, and $\boldsymbol{\epsilon}_t$ is a vector of shocks. $\mathbf{G}$ is the policy matrix, and $\mathbf{H}$ is the state transition matrix. $\mathbf{M}$ is diagonal and contains the standard deviations of the shocks.

$\mathbf{y}_t$ is partitioned into observed and unobserved controls, $\mathbf{y}_t = (\mathbf{y}_{1,t}, \mathbf{y}_{2,t})$. The observed control variables are related to the control variables by the equation

$$\mathbf{y}_{1,t} = \mathbf{D}\mathbf{y}_t$$

where $\mathbf{D}$ is a selection matrix. Only observed control variables play a role estimation. The number of observed control variables must be the same as the number of state equations that include shocks.

You specify a model of the form (9)–(10) to `dsge`. Many models require some manipulation to fit into the structure in (9) and (10); see [DSGE] **intro 4** for details. Postestimation commands `estat policy` and `estat transition` will display the policy and transition matrices in (11) and (12), respectively.

References

Canova, F. 2007. *Methods for Applied Macroeconomic Research*. Princeton, NJ: Princeton University Press.

DeJong, D. N., and C. Dave. 2011. *Structural Macroeconometrics*. 2nd ed. Princeton, NJ: Princeton University Press.

Ljungqvist, L., and T. J. Sargent. 2012. *Recursive Macroeconomic Theory*. 3rd ed. Cambridge, MA: MIT Press.

Taylor, J. B. 1993. Discretion versus policy rules in practice. *Carnegie-Rochester Conference Series on Public Policy* 39: 195–214.

Woodford, M. 2003. *Interest and Prices: Foundations of a Theory of Monetary Policy*. Princeton, NJ: Princeton University Press.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[TS] **forecast** — Econometric model forecasting

[TS] **irf** — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs

[TS] **sspace** — State-space models

Title

intro 2 — Learning the syntax

Description Remarks and examples Also see

Description

In this introduction, we demonstrate how to specify a DSGE model using the `dsge` command. We focus on two unique aspects of DSGEs that must be considered when writing the syntax—writing the system of equations and identifying each type of variable within those equations.

Remarks and examples

Remarks are presented under the following headings:

Preview of dsge syntax

If you have not read [DSGE] **intro 1**, we recommend that you read it first. In particular, see *Structural and reduced forms of DSGE models* in [DSGE] **intro 1**, where we discuss the structural form of a DSGE that is required by `dsge`. Below we assume that your model has this structural form, and we discuss the syntax for specifying such a model with the `dsge` command.

As an example, if we wanted to fit the DSGE model

$$
\begin{aligned}
p_t &= \beta E_t(p_{t+1}) + \kappa y_t \\
y_t &= E_t(y_{t+1}) - (r_t - E_t(p_{t+1}) - \rho z_t) \\
\beta r_t &= p_t + \beta u_t \\
z_{t+1} &= \rho z_t + \epsilon_{t+1} \\
u_{t+1} &= \delta u_t + \xi_{t+1}
\end{aligned}
$$

we would type

```
. dsge (p        = {beta}*E(F.p) + {kappa}*y)                          ///
       (y        = E(F.y) -(r - E(F.p) - {rho}*z), unobserved)         ///
       ({beta}*r = p + {beta}*u)                                       ///
       (F.z      = {rho}*z, state)                                     ///
       (F.u      = {delta}*u, state)
```

The syntax looks a lot like the equations, but it does require a little explanation. In what follows, we discuss each element of the `dsge` syntax that you will need to specify the equations of your DSGE model.

In [DSGE] **intro 1**, we told you that three types of variables—control variables, state variables, and shocks—appear in DSGE models. We will demonstrate how to specify equations involving each of these types of variables. We will also show you how to specify the required types of equations that are linear in these variables and nonlinear in the parameters.

Specifying the system of equations

For a DSGE model, we specify an equation for each control variable and an equation for each state variable. The equations are bound by parentheses. Therefore, the basic structure of the `dsge` command is

```
. dsge (eq1)              ///
       (eq2)              ///
       ..., ...
```

where *eq1* and *eq2* will be replaced with the syntax representing one of the equations in our model. We use ... to indicate that we can include more than two equations as well as options.

The basic form of an equation within the parentheses is

term = *term* [+ *term* [+ *term* [...]]]

where each *term* includes a variable name. The variable name may be preceded by a parameter or nonlinear combination of parameters. For instance, a valid equation might look something like

```
(y = {kappa}*z)
```

or

```
(y = {kappa}*z + {kappa}*{beta}*x)
```

or

```
(1/{beta}*y = {gamma}*z + x)
```

We will explain this later. For now, simply note that these equations are written in `dsge` syntax much like we would write the math, but the parameters we want to estimate are offset with braces, `{}`. If you have used any other Stata commands that work with substitutable expressions, you may recognize this notation. In fact, `dsge` uses a special form of substitutable expressions.

Before we discuss how to use `dsge`'s substitutable expressions, we will focus on how to specify each type of variable.

Control variables

Control variables can be modeled as a function of other control variables, expectations of the future value of control variables, and state variables. The equations for control variables do not include shocks.

We continue from the basic `dsge` command in the previous section, which was

```
. dsge (eq1)              ///
       (eq2)              ///
       ..., ...
```

If we have an observed control variable y, and for *eq1*, we want to specify that y_t be modeled as a function of z_t and x_t, we can type this equation in our `dsge` command as

```
. dsge (y = pexp*z + pexp*x)          ///
  (eq2)                               ///
  ..., ...
```

where *pexp* is a possibly nonlinear expression of parameters in each term.

Control variables can be observed or unobserved. Because we did not include any options within this set of parentheses, y is assumed to be an observed control variable. If y were unobserved, we would add the `unobserved` option as follows:

```
(y = pexp*z + pexp*x, unobserved)
```

Note that each control variable in the model must be included on the left-hand side of one, and only one, equation.

State variables and shocks

To model a state variable, we specify an equation with the one-period lead of that state variable on the left-hand side. On the right-hand side of the equation, we can include state variables, control variables, expectations of the future value of control variables, and shocks.

When we specify an equation for a state variable, we include the `state` option within the parentheses defining the equation.

Continuing with the syntax above, suppose *eq2* is an equation for a state variable x, and we model x_{t+1} as a function of x_t. We expand our `dsge` command to

```
. dsge (y = pexp*z + pexp*x)          ///
  (F.x = pexp*x, state)               ///
  ..., ...
```

We used the `F.` lead operator to specify the one-period lead of x as `F.x`; see [U] **11.4.4 Time-series varlists**. However, notice that the full list of time-series operators is not available here. We can specify only equations for one-period leads of state variables; we could not replace `F.x` with `F2.x` in the equation above. Note that this restriction does not limit the types of models we can fit; see [DSGE] **intro 4c**.

By default, the equation for a state variable includes an unobserved shock. However, equations for state variables are not required to include a shock. Within the system of equations, the number of shocks should be equal to the number of observed control variables. If we did not wish to include a shock in the equation for x_{t+1}, we could add the `noshock` option,

```
(F.x = pexp*x, state noshock)
```

Note that the one-period lead of each state variable must be included on the left-hand side of one, and only one, equation.

Expectations of future values of control variables

Expectations of the one-period lead of control variables can appear in equations for both control and state variables. Mathematically, we write these expectations as $E_t()$. For instance, we write the expectation of y in time $t+1$ as $E_t(y_{t+1})$. In `dsge`, we write this expectation as `E(F.y)`, where `E()` represents an expectation and `F.y` is the one-period lead of y. Other than using the special `E()` notation, expectations are included in the model in the same way as variables.

If we model y_t as a function of $E_t(y_{t+1})$ in addition to z_t and x_t, we can expand our previous `dsge` command to

```
. dsge (y = pexp*E(F.y) + pexp*z + pexp*x)          ///
  (F.x = pexp*x, state)                             ///
  ..., ...
```

Expectations in `E()` are strictly for one-period leads of control variables. You cannot, for instance, use `E(F2.y)` to include the expectation of y in time $t+2$ in the model. This does not prevent you, however, from including such terms in your model. See [DSGE] **intro 4d** for details of fitting models including expectations of control variables more than one period in the future.

Specifying parameters using substitutable expressions

At this point, we know how to specify each type of variable that may arise in our DSGE model. We now turn to specifying the parameters that we want to estimate.

Recall that the basic form of an equation is

term = *term* [+ *term* [+ *term* [...]]]

where each *term* includes a variable name. The variable name may be preceded by a parameter or a nonlinear combination of parameters. For *term*s on the right-hand side of the equation, variable names can also be followed by a parameter specification.

Observed control variables are variables in your dataset. Unobserved control variables and state variables are not variables in your dataset.

We specify the *term*s using a special type of substitutable expressions that we call scalar substitutable expressions. In scalar substitutable expressions, parameters are enclosed in braces, `{}`, and may enter the model either linearly or nonlinearly. The restriction that each term includes only one variable implies that the variables must enter the equation linearly with these scalar substitutable expressions.

If our model for y in the equation above is

$$y_t = \beta E_t(y_{t+1}) + \kappa z_t + \gamma x_t$$

we could extend our previous `dsge` command as follows:

```
. dsge (y = {beta}*E(F.y) + {kappa}*z + {gamma}*x)   ///
  (F.x = exp*x, state)                              ///
  ..., ...
```

However, the equation for y need not be linear in the parameters. If instead we wanted to model y as

$$y_t = (1/\beta)E_t(y_{t+1}) + \kappa z_t + (\gamma/\beta)x_t$$

we would change our command to

```
. dsge (y = 1/{beta}*E(F.y) + {kappa}*z + ({gamma}/{beta})*x)  ///
  (F.x = exp*x, state)                                        ///
  ..., ...
```

We can even include parameters on the left-hand side of the equation. For instance, we could model y as

$$(1/\beta)y_t = E_t(y_{t+1}) + \kappa z_t + (\gamma/\beta)x_t$$

and change our `dsge` command to

```
. dsge ((1/{beta})*y = E(F.y) + {kappa}*z + ({gamma}/{beta})*x)  ///
  (F.x = exp*x, state)                                          ///
  ..., ...
```

Finally, there is an extension to the basic form of the equation that we should mention. Sometimes, it is more convenient to write an equation so that a scalar value or a parameter or a nonlinear combination of parameters is multiplied by a linear combination of terms. The `dsge` command allows this. For instance, instead of

$$y_t = (1/\beta)E_t(y_{t+1}) + (\gamma/\beta)x_t + \kappa z_t$$

we can write

$$y_t = (1/\beta)\{E_t(y_{t+1}) + \gamma x_t\} + \kappa z_t$$

Similarly, we could write this equation in the `dsge` command as

```
(y = (1/{beta}) * (E(F.y) + {gamma}*x) + {kappa}*z)
```

Now you know the general syntax of `dsge` and are ready to fit your own DSGE model with `dsge`.

For examples of fitting classic DSGE models using this syntax, see [DSGE] **intro 3**. If you find some of the rules of this syntax too restrictive for your model—say you want a shock in an equation for a control variable, or you need to include the lag of a state variable—see [DSGE] **intro 4** for examples of rewriting these types of models so that they can be fit using `dsge`.

For more details on the `dsge` syntax, see [DSGE] **dsge**.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **intro 3** — Classic DSGE examples

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 3 — Classic DSGE examples

Description

In this entry, we present three classic DSGE examples—the New Keynesian model, the New Classical model, and the financial frictions model.

Remarks and examples

In [DSGE] **intro 3a**, [DSGE] **intro 3b**, and [DSGE] **intro 3c**, we fit simple variants of common DSGE models. Through these examples, we demonstrate model solution, estimation, and interpretation.

[DSGE] **intro 3a** demonstrates how to fit a New Keynesian model. In this example, we interpret structural parameters, policy matrix parameters, and state transition matrix parameters. We also predict values of both observed control variables and unobserved states.

[DSGE] **intro 3b** illustrates how to solve a New Classical model and plot the IRFs to compare the model's theoretical predictions under different sets of parameter values.

[DSGE] **intro 3c** fits a financial frictions model. In this example, we also estimate parameters of the policy matrix and evaluate the IRFs.

Also see

[DSGE] **intro 4** — Writing a DSGE in a solvable form

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

Title

intro 3a — New Keynesian model

Description Remarks and examples Also see

Description

This introduction estimates and interprets the parameters of a simple New Keynesian model. In this entry, we demonstrate how to constrain parameters in the model and how to interpret structural parameters, policy matrix parameters, and state transition matrix parameters. We also predict values of both observed control variables and unobserved states.

Remarks and examples

Remarks are presented under the following headings:

The model

Equations (1)–(5) specify a canonical New Keynesian model of inflation p_t, the output gap x_t, and the interest rate r_t. These are the linearized equations; the model's nonlinear equations are similar to those in *How to write down a DSGE* in [DSGE] **intro 1**.

$$p_t = \beta E_t(p_{t+1}) + \kappa x_t \tag{1}$$
$$x_t = E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \tag{2}$$
$$r_t = \psi p_t + u_t \tag{3}$$
$$u_{t+1} = \rho_u u_t + \epsilon_{t+1} \tag{4}$$
$$g_{t+1} = \rho_g g_t + \xi_{t+1} \tag{5}$$

Equation (1) specifies inflation as a linear combination of expected future inflation and the output gap. Equation (2) specifies the output gap as a linear combination of the expected future output gap, the real interest rate, and a state variable g_t. Equation (3) specifies the interest rate as a linear combination of inflation and a state variable u_t. The state variables are modeled as first-order autoregressive processes. The state variable u_t is the deviation of r_t from its equilibrium value of ψp_t. The state variable g_t is also the deviation of x_t from its equilibrium value.

Three of the parameters have structural interpretation. The parameter κ is known as the slope of the Phillips curve and is predicted to be positive. The parameter β is the discount factor that represents the degree to which agents discount the future relative to the current period. The parameter ψ measures the degree to which interest rates react to movements in inflation.

Parameter estimation

Not all model parameters are identified. We constrain β to be 0.96, a common value in the literature. The remaining parameters are identified.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. constraint 1 _b[beta]=0.96

. dsge (p = {beta}*E(F.p) + {kappa}*x)
>       (x = E(F.x) -(r - E(f.p) - g), unobserved)
>       (r = {psi}*p + u)
>       (F.u = {rhou}*u, state)
>       (F.g = {rhog}*g, state),
>     from(psi=1.5)  constraint(1)
(setting technique to bfgs)
Iteration 0:   log likelihood = -5736.4646
Iteration 1:   log likelihood = -1190.0604  (backed up)
Iteration 2:   log likelihood = -960.00953  (backed up)
Iteration 3:   log likelihood = -928.79225  (backed up)
Iteration 4:   log likelihood = -842.40806  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -810.92149  (backed up)
Iteration 6:   log likelihood = -766.17852  (not concave)
Iteration 7:   log likelihood = -760.42901
Iteration 8:   log likelihood = -756.77337
Iteration 9:   log likelihood = -754.05236
Iteration 10:  log likelihood = -753.58052
Iteration 11:  log likelihood = -753.57142
Iteration 12:  log likelihood = -753.57131
Iteration 13:  log likelihood = -753.57131
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
 ( 1)  [/structural]beta = .96
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.96	(constrained)				
kappa	.0849633	.0287693	2.95	0.003	.0285764	.1413502
psi	1.943007	.2957876	6.57	0.000	1.363274	2.52274
rhou	.7005481	.0452603	15.48	0.000	.6118395	.7892568
rhog	.9545256	.0186424	51.20	0.000	.9179872	.991064
sd(e.u)	2.318207	.3047443			1.720919	2.915495
sd(e.g)	.5689901	.0982974			.3763308	.7616494

The slope of the Phillips curve, kappa, is estimated to be positive. The coefficient on inflation in the interest rate equation is estimated to be almost 1.94, meaning that interest rates are expected to rise almost two for one with increases in inflation.

Policy and transition matrices

Elements of the policy matrix represent the response of a control variable to a one-unit increase in a state variable.

```
. estat policy
Policy matrix
```

		Coef.	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
p							
	u	-.4172517	.039361	-10.60	0.000	-.4943978	-.3401056
	g	.9678148	.2777441	3.48	0.000	.4234464	1.512183
x							
	u	-1.608212	.4050524	-3.97	0.000	-2.4021	-.8143236
	g	.9529166	.481391	1.98	0.048	.0094076	1.896426
r							
	u	.1892771	.059166	3.20	0.001	.0733139	.3052402
	g	1.880471	.2615988	7.19	0.000	1.367746	2.393195

An increase in u decreases the extent to which the inflation is above its short-run equilibrium value. This change decreases the output gap and increases interest rate.

An increase in g increases the extent to which the inflation is above its short-run equilibrium value. This change also increases output gap and interest rates.

Because the states are uncorrelated with each other in this example, the elements of the state transition matrix are just the persistence parameters in the model.

```
. estat transition
Transition matrix of state variables
```

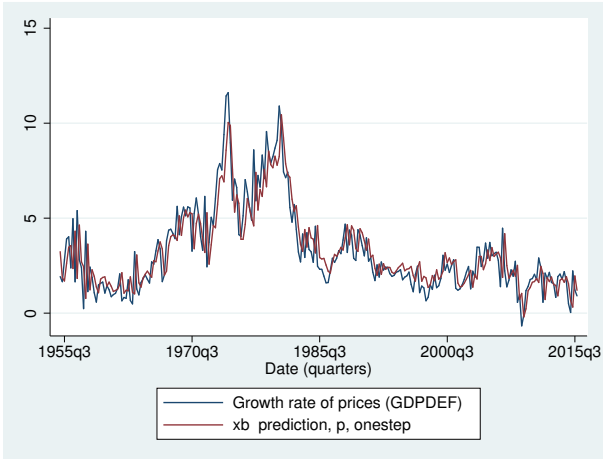
		Coef.	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
F.u							
	u	.7005481	.0452603	15.48	0.000	.6118395	.7892568
	g	1.67e-16	9.03e-12	0.00	1.000	-1.77e-11	1.77e-11
F.g							
	u	0	(omitted)				
	g	.9545256	.0186424	51.20	0.000	.9179872	.991064

See [DSGE] **intro 4g** for an example in which the state variables depend on each other. The states can also depend on each other when a state variable is specified to depend on control variables, as in [DSGE] **intro 3b**, or when the state vector includes lagged control variables, as in [DSGE] **intro 4a**.

One-step ahead predictions

Predictions after dsge depend on the estimated state-space parameters. Below we obtain one-step ahead predictions for each of the two observed control variables in the model, and we graph the actual and predicted inflation rate.

```
. predict dep*
(option xb assumed; fitted values)
. tsline p dep1, legend(col(1))
```



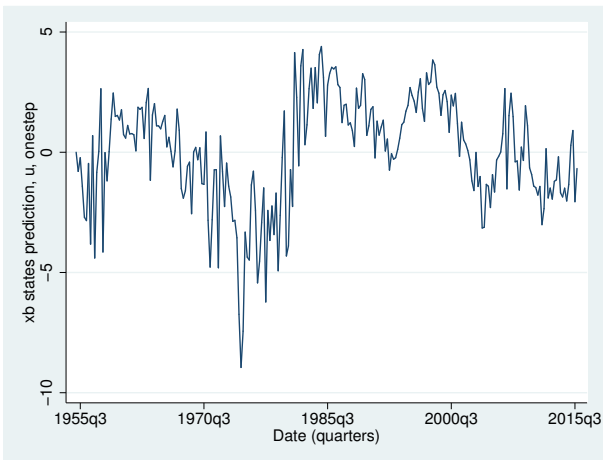
The graph shows that the one-step ahead predictions closely follow realized inflation.

Estimating an unobserved state

The observed control variables are driven by two unobserved state variables. We can use `predict` with the `state` option to estimate the state variables.

Here we estimate the unobserved state `u` and plot it.

```
. predict state1, state
. tsline state1
```



The loose monetary policy in the mid 1970s where predicted values of u_t are negative and the subsequent Volcker contraction are apparent in this plot.

Also see

[DSGE] **intro 1** — Introduction to DSGEs and dsge

[DSGE] **intro 3b** — New Classical model

[DSGE] **intro 3c** — Financial frictions model

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

Title

intro 3b — New Classical model

Description Remarks and examples Reference Also see

Description

In this example, we solve a New Classical model similar to the one in King and Rebelo (1999). We also demonstrate how to compare a model's theoretical predictions under different parameter values using IRFs.

Remarks and examples

Remarks are presented under the following headings:

The model

In this model, output, consumption, investment, employment, and other variables are driven by state variables linked to production and demand. The model is similar to the one in King and Rebelo (1999) and is referred to as a real business cycle model.

The nonlinear form of the model is

$$\frac{1}{C_t} = \beta E_t\left\{\frac{1}{C_{t+1}}(1 + R_{t+1} - \delta)\right\} \tag{1a}$$

$$H_t^{\eta} = \frac{W_t}{C_t} \tag{2a}$$

$$Y_t = C_t + X_t + G_t \tag{3a}$$

$$Y_t = K_t^{\alpha}(Z_t H_t)^{1-\alpha} \tag{4a}$$

$$W_t = (1-\alpha)\frac{Y_t}{H_t} \tag{5a}$$

$$R_t = \alpha\frac{Y_t}{K_t} \tag{6a}$$

$$K_{t+1} = (1-\delta)K_t + X_t \tag{7a}$$

The first equation specifies consumption C_t as a function of expected future consumption and the expected future interest rate $E_t(R_{t+1})$. Equation (2a) specifies labor hours H_t as a function of the wage W_t and consumption; it is a labor supply equation. Equation (3a) is the national income accounting identity for a closed economy, specifying output Y_t as the sum of consumption, investment X_t, and government spending G_t. Equation (4a) is a production function that specifies output as a function of labor input H_t, capital input K_t, and productivity Z_t. Equations (5a) and (6a) specify labor demand and capital demand, respectively. Equation (7a) specifies the equation for capital accumulation. The model is completed when we add state transition equations for Z_t and G_t. These state transition equations are conventionally specified after the model has been linearized.

The linearized form of the model is

$$
\begin{aligned}
c_t &= E_t(c_{t+1}) - (1 - \beta + \beta\delta)E_t(r_{t+1}) \\
\eta h_t &= w_t - c_t \\
\phi_1 x_t &= y_t - \phi_2 c_t - g_t \\
y_t &= (1-\alpha)(z_t + h_t) + \alpha k_t \\
w_t &= y_t - h_t \\
r_t &= y_t - k_t \\
k_{t+1} &= \delta x_t + (1-\delta)k_t \\
z_{t+1} &= \rho_z z_t + \epsilon_{t+1} \\
g_{t+1} &= \rho_g g_t + \xi_{t+1}
\end{aligned}
$$

The model has six control variables and three state variables. Two of the state variables, z_{t+1} and g_{t+1}, are modeled as first-order autoregressive processes. The state equation for k_{t+1} depends on the current value of a control variable, namely, x_t.

Solving the model

The `solve` option of `dsge` places the model in state-space form without estimating parameters; it is similar to `iterate(0)` but is faster because it does not calculate standard errors. Using `solve` for different parameter values of your model is a useful way to explore the model's theoretical properties.

The parameter values used here are similar to those used in King and Rebelo (1999). Each has an interpretation. (1 − `alpha`) is labor's share of national income. `delta` is the depreciation rate of capital. `eta` is the slope of the labor supply curve. `phi1` and `phi2` are share parameters related to investment's share of national income and consumption's share of national income, respectively. `rhoz` and `rhog` are autoregressive parameters on the state variables.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)

. dsge (c = E(F.c) - (1-{beta}+{beta}*{delta})*E(F.r), unobserved)
>      ({eta}*h = w - c, unobserved)
>      ({phi1}*x = y - {phi2}*c - g, unobserved)
>      (y = (1-{alpha})*(z+h) + {alpha}*k)
>      (w = y - h, unobserved)
>      (r = y - k, unobserved)
>      (F.k = {delta}*x+ (1-{delta})*k, state noshock)
>      (F.z = {rhoz}*z, state)
>      (F.g = {rhog}*g, state),
>      from(beta=0.96 eta=1 alpha=0.3 delta=0.025 phi1=0.2 phi2=0.6 rhoz=0.8
>      rhog=0.3) solve noidencheck
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1957.0261
```

y	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.96	.	.	.	.	.
delta	.025	.	.	.	.	.
eta	1	.	.	.	.	.
phi1	.2	.	.	.	.	.
phi2	.6	.	.	.	.	.
alpha	.3	.	.	.	.	.
rhoz	.8	.	.	.	.	.
rhog	.3	.	.	.	.	.
sd(e.z)	1	.			.	.
sd(e.g)	1	.			.	.

```
Note: Skipped identification check.
Note: Model solved at specified parameters.
```

The `solve` option solves the model at the specified values in `from()`. We skip the identification check with `noidencheck`. Simply solving the model does not involve any reference to the data or any estimation. Still we can explore what these parameters imply.

Policy and transition matrices

After solving, we can use many of the postestimation commands, though standard errors will be missing throughout.

The state transition matrix shows how the state vector in the next period is related to the state vector in the current period. Some state variables are specified as first-order autoregressive processes, and their transition equations will simply repeat information that is already available in the estimation table. However, if any state variable equation contains control variables, then that state variable's transition equation will depend on the other state variables.

```
. estat transition
Transition matrix of state variables
```

	Coef.	Delta-method Std. Err.	z	P>|z|	[95% Conf.	Interval]
F.k						
k	.9256785	.	.	.	.	.
z	.1078282	.	.	.	.	.
g	-.1070547	.	.	.	.	.
F.z						
k	0	(omitted)				
z	.8	.	.	.	.	.
g	2.22e-16	.	.	.	.	.
F.g						
k	0	(omitted)				
z	0	(omitted)				
g	.3	.	.	.	.	.

The value of the state variables z and g in the next period depends only on their value in the current period, but the value of the capital stock k in the next period depends on the current value of all three state variables. This feature means that, for example, a shock to the z state variable has two effects: it increases future values of z, because z is autoregressive, but it also increases future values of k. Interrelationships among the state variables can generate more interesting patterns in the IRFs than the AR(1) dynamics that we saw in [DSGE] **intro 3a**.

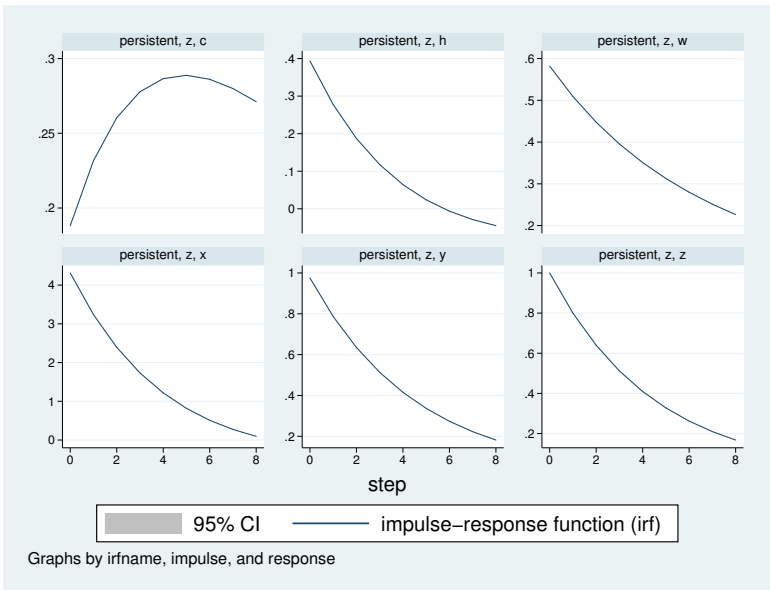
Impulse responses

One way to compare two parameter sets is to graph the impulse response of model variables to a shock under each parameter set. We first set the impulse–response file with `irf set` and then add impulse responses named `persistent` to the file with `irf create`.

```
. irf set rbcirf
(file rbcirf.irf created)
(file rbcirf.irf now active)

. irf create persistent
(file rbcirf.irf updated)
```

The response of model variables to a shock to z is graphed by typing

```
. irf graph irf, irf(persistent) impulse(z) response(y c x h w z)
> byopts(yrescale)
```



Each graph is labeled with the IRF name, the impulse variable, and the response variable. For instance, the top-left graph shows the response of consumption to a shock to the z_t state variable. The bottom-right graph shows the response of the state variable z_t itself. The state is persistent, which is not surprising: we set the autoregressive parameter in the z_t equation to 0.8.

In the top-left graph, we see that consumption c rises over time before returning to steady state. The time unit is quarters, so a value of about 0.27, 4 periods after the shock, indicates that consumption

is 0.27% above its steady-state value one year after the shock. Hours worked `h` are shown in the top center graph and rise initially before falling below steady state. The real wage `w`, output `y`, and investment `x` all rise.

Sensitivity analysis

The responses of variables to a shock to `z` are persistent. Some variables, like consumption and the wage, show dynamics beyond the simple autoregressive behavior of `z` itself. How much of this behavior is due to the intrinsic propagation mechanisms of the model, and how much is due to the persistence of `z`?

To evaluate this, we rerun the `dsge` command. This time we set the persistence of `z` to a small value of 0.01.

```
. dsge (c = E(F.c) - E(F.r), unobserved)
>      ({eta}*h = w - c, unobserved)
>      ({phi1}*x = y - {phi2}*c - g, unobserved)
>      (y = (1-{alpha})*(z+h) + {alpha}*k)
>      (w = y - h, unobserved)
>      (r = y - k, unobserved)
>      (F.k = {delta}*x+ (1-{delta})*k, state noshock)
>      (F.z = {rhoz}*z, state)
>      (F.g = {rhog}*g, state),
>      from(eta=1 alpha=.3 delta=0.025 phi1=0.2 phi2=0.6 rhoz=0.01 rhog=0.3)
>      solve noidencheck
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood =  -2002.837
```

y	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
eta	1	.	.	.	.	.
phi1	.2	.	.	.	.	.
phi2	.6	.	.	.	.	.
alpha	.3	.	.	.	.	.
delta	.025	.	.	.	.	.
rhoz	.01	.	.	.	.	.
rhog	.3	.	.	.	.	.
sd(e.z)	1	.			.	.
sd(e.g)	1	.			.	.

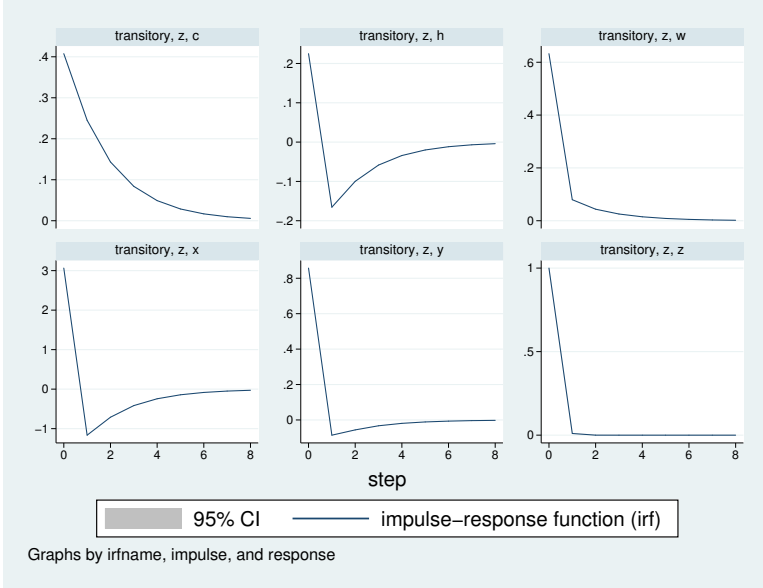
```
Note: Skipped identification check.
Note: Model solved at specified parameters.
```

The only change in the parameter set is that `rhoz` has been set to 0.01 from its earlier setting of 0.9. We can add the impulse responses of this model to the `irf` file with the name `transitory`,

```
. irf create transitory, replace
irfname transitory not found in rbcirf.irf
(file rbcirf.irf updated)
```

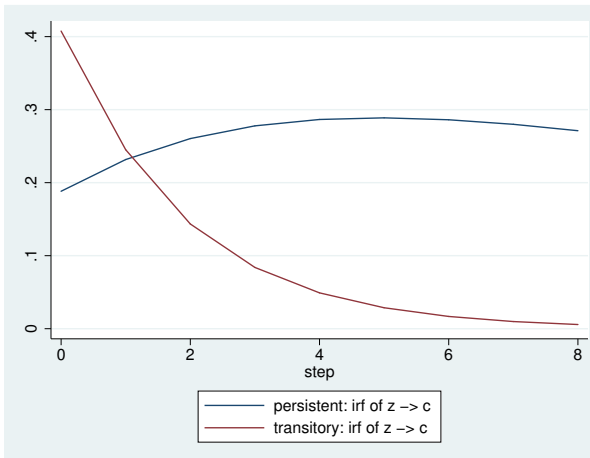
and graph them.

```
. irf graph irf, irf(transitory) impulse(z) response(y c x h w z)
> byopts(yrescale)
```



Model variables are much less persistent. All the impulse responses have changed shape. We can use `irf ograph` to overlay the IRF for a variable under the two calibrations. This way we can view the differences across calibrations directly.

```
. irf ograph (persistent z c irf) (transitory z c irf)
```



When the shock itself is persistent, consumption responds persistently. When the shock is transitory, consumption returns to its steady-state value quickly.

Reference

King, R. G., and S. T. Rebelo. 1999. Resuscitating real business cycles. In *Handbook of Macroeconomics: Volume 1A*, ed. J. B. Taylor and M. Woodford, 927–1007. New York: Elsevier.

Also see

[DSGE] **intro 1** — Introduction to DSGEs and dsge

[DSGE] **intro 3a** — New Keynesian model

[DSGE] **intro 3c** — Financial frictions model

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

Title

intro 3c — Financial frictions model

Description Remarks and examples Also see

Description

This introduction estimates and interprets the parameters of a model that incorporates financial frictions. The model is an extension of the one in [DSGE] **intro 3a**.

Remarks and examples

Remarks are presented under the following headings:

The model

Equations (1)–(7) specify a model of financial frictions. The simplest such model places a wedge between two interest rates: the safe interest rate set by the central bank and the market interest rate that consumers and producers use.

$$\pi_t = \beta E_t \pi_{t+1} + \kappa x_t \tag{1}$$
$$x_t = E_t x_{t+1} - (i_t - E_t \pi_{t+1} - g_t) \tag{2}$$
$$r_t = \psi \pi_t + u_t \tag{3}$$
$$i_t = \chi r_t + e_t \tag{4}$$
$$g_{t+1} = \rho_g g_t + \xi_{t+1} \tag{5}$$
$$u_{t+1} = \rho_u u_t + \epsilon_{t+1} \tag{6}$$
$$e_{t+1} = \rho_e e_t + \eta_{t+1} \tag{7}$$

These are the linearized equations. The nonlinear equations are similar to those in *How to write down a DSGE* in [DSGE] **intro 1**.

This model is an extension to the one worked in [DSGE] **intro 1**. It has an additional equation for the interest rate spread. As before, (1) specifies the inflation equation (a Phillips curve), (2) specifies the output gap equation (an Euler equation), and (3) specifies the equation for the safe interest rate (a Taylor rule). Equation (3) can be thought of as an equation that specifies the safe interest rate r_t. The new element is (4), which specifies an equation for the market interest rate i_t that enters the output gap equation. The market interest rate i_t is a function of the safe interest rate and a state variable e_t. The state variable e_t controls the interest rate spread and can be thought of as representing the state of the financial system. A large realization of e_t represents a large interest rate spread, indicating financial distress.

Parameter estimation

We estimate the parameters of the model in (1)–(7) using U.S. data on the Federal funds rate (safe rate set by the central bank), the high-grade corporate bond interest rate (a measure of market interest rates), and the inflation rate.

As is typical in these models, we constrain the parameter `beta` to 0.96.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. constraint 1 _b[beta]=0.96
. dsge (p = {beta}*E(F.p) + {kappa}*x)
>      (x = E(F.x) -(i - E(f.p) - g), unobserved)
>      (i = {chi}*r + e)
>      (r = {psi}*p + u)
>      (F.e = {rhoe}*e, state)
>      (F.u = {rhou}*u, state)
>      (F.g = {rhoz}*g, state),
>      from(psi=2 chi=0.8) constraint(1)
(setting technique to bfgs)
Iteration 0:   log likelihood = -4780.2037
Iteration 1:   log likelihood = -1731.5956  (backed up)
Iteration 2:   log likelihood = -1315.8819  (backed up)
Iteration 3:   log likelihood = -1161.0796  (backed up)
Iteration 4:   log likelihood = -1115.2257  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1069.1254  (backed up)
Iteration 6:   log likelihood = -1016.9948  (not concave)
Iteration 7:   log likelihood = -980.93826  (not concave)
Iteration 8:   log likelihood = -922.65912  (not concave)
Iteration 9:   log likelihood = -914.78652  (not concave)
Iteration 10:  log likelihood =  -908.4795  (not concave)
Iteration 11:  log likelihood = -905.35477  (not concave)
Iteration 12:  log likelihood = -902.51101  (not concave)
Iteration 13:  log likelihood = -900.48635  (not concave)
Iteration 14:  log likelihood = -899.17535  (not concave)
Iteration 15:  log likelihood = -897.88451  (not concave)
Iteration 16:  log likelihood = -895.03731
Iteration 17:  log likelihood = -891.34894
Iteration 18:  log likelihood = -887.41441
Iteration 19:  log likelihood = -882.82925  (not concave)
Iteration 20:  log likelihood = -882.73272
Iteration 21:  log likelihood = -882.20988
Iteration 22:  log likelihood = -882.13462
Iteration 23:  log likelihood = -882.13198
Iteration 24:  log likelihood = -882.13195
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -882.13195
 ( 1)  [/structural]beta = .96
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.96	(constrained)				
kappa	.0503238	.0184922	2.72	0.007	.0140796	.0865679
chi	.9067346	.1249748	7.26	0.000	.6617885	1.151681
psi	6.332415	2.56766	2.47	0.014	1.299893	11.36494
rhoe	.8478235	.0301215	28.15	0.000	.7887865	.9068605
rhou	.8153493	.0332519	24.52	0.000	.7501768	.8805218
rhoz	.9861866	.0099767	98.85	0.000	.9666327	1.005741
sd(e.e)	.8857022	.1343			.622479	1.148925
sd(e.u)	7.16076	2.933846			1.410528	12.91099
sd(e.g)	.3911864	.0358355			.3209501	.4614226

The persistence of the financial shock `rhoe` is estimated to be 0.85. The slope of the Phillips curve, `kappa`, is somewhat flatter in this model than in the one in [DSGE] **intro 1**. The coefficient on inflation in the interest rate equation is 6.3, and indicates that the central bank increases interest rates much more than one for one in response to movements in inflation.

Policy and transition matrices

We can read off the impact effect of shocks using the policy matrix. The response to the financial shock `e` will be of most interest.

```
. estat policy
Policy matrix
```

		Delta-method					
		Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
p							
	e	-.1832602	.0740955	-2.47	0.013	-.3284848	-.0380357
	u	-.1584184	.0641356	-2.47	0.014	-.2841219	-.032715
	g	.2096325	.1003838	2.09	0.037	.0128838	.4063812
x							
	e	-.6776679	.3498926	-1.94	0.053	-1.363445	.0081091
	u	-.6839457	.2783497	-2.46	0.014	-1.229501	-.1383903
	g	.2218675	.1284759	1.73	0.084	-.0299406	.4736756
i							
	e	-.0522472	.040321	-1.30	0.195	-.1312748	.0267805
	u	-.0028753	.0041244	-0.70	0.486	-.0109589	.0052083
	g	1.203672	.0979354	12.29	0.000	1.011722	1.395622
r							
	e	-1.16048	.1361163	-8.53	0.000	-1.427263	-.8936968
	u	-.003171	.0045309	-0.70	0.484	-.0120515	.0057094
	g	1.32748	.2224889	5.97	0.000	.8914098	1.76355

Importantly, the financial shock causes `r` to fall by more than `i`, indicating an increase in the interest rate spread. Inflation also falls, as does the output gap. It is the fall in the inflation rate that causes the central bank's interest rate to fall on impact of the shock.

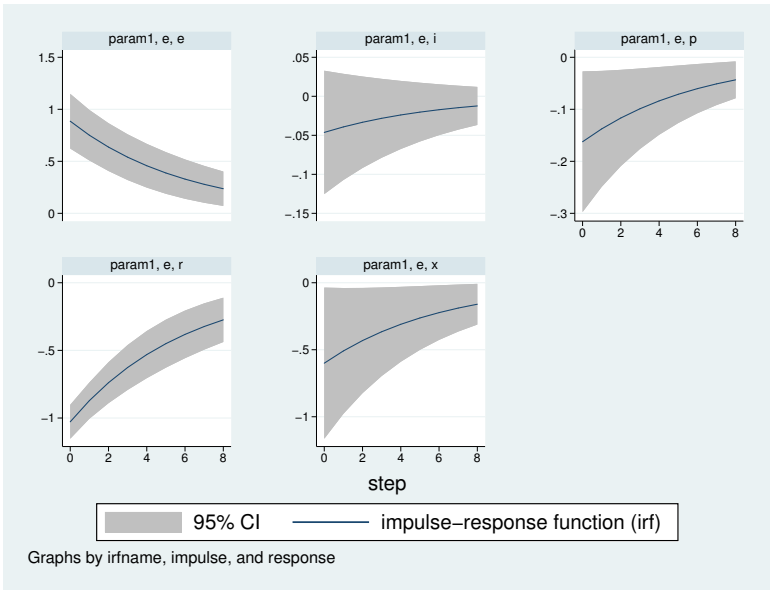
Impulse responses

We use the `irf set` command to set `finirf.irf` as the active IRF file and then use `irf create` to create and store the impulse responses under the name `param`.

```
. irf set finirf
(file finirf.irf created)
(file finirf.irf now active)

. irf create param1
(file finirf.irf updated)
```

Finally, we graph the impulse response to a financial shock.

```
. irf graph irf, irf(param1) impulse(e) response(e x p i r) byopts(yrescale)
```



The top middle panel shows the effect of the shock to the `e` equation on `e` itself. The shock represents a persistent increase in the spread between the two interest rates. The increased interest rate spread causes both the output gap and inflation rate to fall. The safe interest rate `r` falls dramatically, and the combined effect of the increased spread `e` and reduced safe interest rate `r` causes the market interest rate `i` to be little changed.

Also see

[DSGE] **intro 1** — Introduction to DSGEs and dsge

[DSGE] **intro 3a** — New Keynesian model

[DSGE] **intro 3b** — New Classical model

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

Title

intro 4 — Writing a DSGE in a solvable form

Description Remarks and examples Also see

Description

Many DSGE models, when written in their natural representation based on economic theory, include problematic terms that do not fit into the algebraic form required to solve a DSGE model. This entry shows how to accommodate these problematic terms by defining a new state variable or a new control variable and then rewriting the equations so that they have the required form.

We assume that you are already familiar with the elements of a DSGE; see [DSGE] **intro 1**.

Remarks and examples

Remarks are presented under the following headings:

Introduction

We can estimate the parameters of a DSGE model only if we can solve for its state-space form, and we can solve a DSGE model for its state-space form only if the equations for control variables and state variables have specific structures. Unfortunately, many DSGE models contain problematic terms that do not fit into these structures without some manipulation. Fortunately, the manipulation is easy, once you know the tricks discussed below.

As we discussed in *Structural and reduced forms of DSGE models* in [DSGE] **intro 1**, we can only solve a DSGE whose structural form can be written as

$$\mathbf{A}_0\mathbf{y}_t = \mathbf{A}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{A}_2\mathbf{y}_t + \mathbf{A}_3\mathbf{x}_t \tag{1}$$

$$\mathbf{B}_0\mathbf{x}_{t+1} = \mathbf{B}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{B}_2\mathbf{y}_t + \mathbf{B}_3\mathbf{x}_t + \mathbf{C}\boldsymbol{\epsilon}_{t+1} \tag{2}$$

where $\mathbf{y}_t$ is a vector of control variables, $\mathbf{x}_t$ is a vector of state variables, and $\boldsymbol{\epsilon}_t$ is a vector of shocks. $\mathbf{A}_0$ through $\mathbf{A}_3$ and $\mathbf{B}_0$ through $\mathbf{B}_3$ are matrices of parameters. The entries in $\mathbf{A}_i$ and $\mathbf{B}_j$ are all functions of the structural parameters, which we denote by vector $\boldsymbol{\theta}$. $\mathbf{A}_0$ and $\mathbf{B}_0$ are diagonal matrices, and $\mathbf{A}_2$ has zeros on its diagonal. $\mathbf{C}$ is a selection matrix that determines which state variables are subject to shocks.

Equation (1) specifies the required structure for control variable equations, and (2) specifies the required structure for state variable equations. Per (1), a control can depend on only three types of terms—expected values of control variables in the next period, current values of other control variables, and current values of state variables. All other terms are problematic. Per (2), the values of state variables in the next period can depend on only four types of terms—expected values of control variables in the next period, current values of control variables, current values of state variables, and shocks. All other terms are problematic.

When written based on economic theory, the structural forms of many DSGE models include problematic terms that do not fit into the required form. These problematic terms are handled by defining a new state variable or a new control variable and rewriting the equations to have the required form. We define a new state variable when the problematic term involves an exogenous, also known as a predetermined, variable. We define a new control variable when the problematic term includes a new endogenous variable. The rewriting process replaces the variable in the problematic term with the new state or the new control variable.

Problematic terms usually violate one of the following four restrictions:

1. Equations for control variables may not contain shocks.
2. Equations may not contain lagged control variables.
3. Equations may not contain lagged state variables.
4. Equations may not contain expectations of control variables dated by more than one period ahead.

All of these restrictions can be overcome by adding a new state variable or a new control variable and rewriting the model so that it has the required form. Here are five examples of the four restrictions and solutions to each one.

1. If you have an equation for a control variable that contains a shock, make this shock a new state variable; see *Shocks to a control equation* below.
2. If you have an equation for a control variable that contains a lagged control variable, make the lagged control variable a new state variable; see *Including a lag of a control variable* below.
3. If you have an equation for a state variable that contains a lag of a state variable, make the lagged state variable a new state variable; see *Including a lag of a state variable* below.
4. If you have an equation for a control variable that contains an expectation of a control variable dated by more than one period ahead, make it a new control variable. We use a new control variable instead of a new state variable because the far-future expectation is endogenous, not predetermined. See *Including an expectation of a control dated by more than one period ahead* below for details.
5. If you have an equation for a control variable that contains a second-order lag of a control variable, make the second-order lagged control variable a new state variable. You will also need to create a new state variable for the first-order lag. See *Including a second-order lag of a control variable* below for details.

A nuanced issue not covered by restrictions 1–4 is that all the observed variables in a DSGE must be endogenous control variables. This structure is usually not a problem, because DSGE models are usually generated by theories that treat everything observed as endogenous. Occasionally, however, models include an observed variable that is exogenous. For example, a small, open economy might not be able to affect its real exchange rate. The solution for this problem is to model the observed variable as a control variable that is equal to an exogenous state variable. See *Including an observed exogenous variable* below for details.

For the most part, this entry discusses how to make models with problematic terms fit into the structure of (1) and (2). Note that correlated state variables already fit into this structure. This correlation is modeled by off-diagonal elements in $\mathbf{B_3}$, not by elements in $\mathbf{C}$. See [DSGE] **intro 4g** for an example.

Shocks to a control equation

Sometimes, we only observe a control variable with error. Suppose that the equation for the control variable consumption is

$$c_t = E_t c_{t+1} - r_t + \epsilon_t \tag{3}$$

where c_t is consumption, r_t is the interest rate, and ϵ_t is a shock. Shocks to a control variable are not allowed, so ϵ_t is a problematic term.

The solution entails three steps. First, we define a new state variable $u_t = \epsilon_t$. We define a new state instead of a new control because the shock is exogenous. Second, we write a state equation for u_t. Recall that the state equation specifies tomorrow's state value in terms of today's state value and tomorrow's shock. The state equation for u_t is

$$u_{t+1} = \epsilon_{t+1}$$

Third, we rewrite (3), substituting u_t for ϵ_t. The consumption equation becomes

$$c_t = E_t c_{t+1} - r_t + u_t$$

which is in the required form. In the `dsge` command, this is expressed as

```
. dsge ...                  ///
       (c = E(F.c) - r + u) ///
       (F.u = , state)      ///
       ...
```

See [DSGE] **intro 4a** for a complete example.

Including a lag of a control variable

Models that mix adaptive and rational expectations contain a lag of the control variable and the expected future value of the control variable. Suppose our model had the following equation for the control variable y_t,

$$y_t = \alpha y_{t-1} + (1-\alpha)E_t(y_{t+1}) - r_t \tag{4}$$

where y_t is output and r_t is the interest rate. The problematic term is αy_{t-1} because it involves a lag of the control variable.

The solution entails three steps. First, we define a new state variable $Ly_t = y_{t-1}$. We define a new state instead of new control because the lagged control is predetermined, which makes it exogenous. Second, we write its state equation as

$$Ly_{t+1} = y_t$$

Third, we rewrite (4), substituting Ly_t for y_{t-1},

$$y_t = \alpha L y_t + (1-\alpha)E_t(y_{t+1}) - r_t$$

which is in the required form. To do this using the `dsge` command, we would type

```
. dsge ...                                      ///
       (y = {alpha}*Ly + (1-{alpha})*E(F.y) - r) ///
       (F.Ly = y, state noshock)                ///
       ...
```

See [DSGE] **intro 4b** for a complete example.

Including a lag of a state variable

Lagged state variables are handled in a manner analogous to lagged control variables. Suppose we have an equation for a state variable z_t that contains a lag of itself. For example, this occurs when z_t is a second-order autoregressive process instead of a first-order autoregressive process. Specifically, our model contains the equation

$$z_{t+1} = \rho_1 z_t + \rho_2 z_{t-1} + \epsilon_{t+1} \tag{5}$$

The problematic term is $\rho_2 z_{t-1}$ because it violates the form required by (2).

The solution entails three steps. First, we define a new state variable $Lz_t = z_{t-1}$. We define a new state instead of new control because the lagged state is exogenous. Second, we write its state equation as

$$Lz_{t+1} = z_t$$

Third, we rewrite (5), substituting Lz_t for z_{t-1},

$$z_{t+1} = \rho_1 z_t + \rho_2 Lz_t + \epsilon_{t+1}$$

which is in the required form. In the `dsge` command, we would type

```
. dsge ...                                  ///
       (F.z = {rho1}*z + {rho2}*Lz, state)  ///
       (F.Lz = z, state noshock)            ///
       ...
```

See [DSGE] **intro 4c** for a complete example.

Including an expectation of a control dated by more than one period ahead

Many models include an expectation of a control variable dated by more than one period ahead. For example, suppose that the equation for the control variable consumption growth is

$$c_t = (1-h)w_t + hE_t c_{t+2} + r_t \tag{6}$$

where c_t is consumption growth, w_t is wage growth, and r_t is the interest rate. Only expectations of control variables dated one period ahead are allowed, so $E_t(c_{t+2})$ is a problematic term.

The solution entails three steps. First, we define a new control variable $Fc_t = E_t(c_{t+1})$. The new variable is a control instead of a state because $E_t(c_{t+1})$ is endogenous instead of exogenous. Second, we include this new control equation,

$$Fc_t = E_t(c_{t+1})$$

Third, we substitute Fc_{t+1} for c_{t+2} in (6). The equation becomes

$$c_t = (1-h)w_t + hE_t(Fc_{t+1}) + r_t$$

which is in the required form. Using the `dsge` command, we would type

```
. dsge ...                                  ///
       (c = (1-{h})*w + {h}*E(F.Fc) + r)    ///
       (Fc = E(F.c), unobserved)            ///
       ...
```

See [DSGE] **intro 4d** for a complete example.

Including a second-order lag of a control variable

Higher-order lags require defining more than one new state variable, as we illustrate here.

Equations for the state variable capital frequently involve "time to build," so that the control variable investment made in time t becomes available only as new capital several periods in the future. Suppose it takes three periods to convert investment x_t into capital. We denote current capital by k_t and write this three-period relationship as

$$k_{t+1} = (1-\delta)k_t + x_{t-2} \tag{7}$$

which specifies that the capital stock tomorrow is the sum of investment made two periods ago and the capital today that has not depreciated at rate δ.

The term x_{t-2} is problematic because it is a second-order lag of the state variable x_t, which violates the form required by (2).

The solution entails three steps. First, we define the new state variables $L2x_t = x_{t-2}$ and $Lx_t = x_{t-1}$. Second, we write their state equations as

$$L2x_{t+1} = Lx_t$$

and

$$Lx_{t+1} = x_t$$

Third, we rewrite (7) as

$$k_{t+1} = (1-\delta)k_t + L2x_t$$

With the `dsge` command, we could fit this model as

```
. dsge ...                              ///
       (F.k = (1-{delta})*k + L2x)      ///
       (F.L2x = Lx, state noshock)      ///
       (F.Lx = x, state noshock)        ///
       ...
```

For a complete example, see [DSGE] **intro 4e**.

In general, if the lag $t-k$ appears anywhere in any of your equations, you will need k new state equations.

Including an observed exogenous variable

Sometimes, we want to treat an observed variable as exogenous. Suppose we want to treat the changes in an observed exchange rate e_t as exogenous. The problem is that all the observed variables in a DSGE model must be modeled as endogenous control variables. The solution is to define a control variable that is equal to a state variable that models the exogenous process. So we could set the observed exchange rate e_t to equal the exchange rate state, es_t,

$$e_t = es_t$$

and specify, say, a first-order autoregressive process for es_t,

$$es_{t+1} = \rho_e es_t + \eta_{t+1}$$

To fit this type of model using the `dsge` command, we would type

```
. dsge ...                              ///
       (e = es)                         ///
       (F.es = {rhoe}*es, state)        ///
       ...
```

See [DSGE] **intro 4f** for a complete example.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 3** — Classic DSGE examples

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

Title

intro 4a — Specifying a shock on a control variable

Description Remarks and examples Also see

Description

A shock to a control variable is problematic because it does not fit into the form of a structural model that is required to solve for the state-space form. This entry shows how to solve this problem by defining a new state variable and rewriting the equations.

Remarks and examples

Remarks are presented under the following headings:

The model

Equations (1)–(3) specify a model of consumption growth and growth in hours worked. The equation for consumption growth is subject to an additive shock.

$$c_t = (1-h)w_t + hE_tc_{t+1} + \epsilon_t \tag{1}$$
$$n_t = w_t - \gamma c_t \tag{2}$$
$$w_{t+1} = \rho w_t + \xi_{t+1} \tag{3}$$

Equation (1) specifies that consumption growth c_t is a linear combination of wage growth w_t, expected future consumption growth E_tc_{t+1}, and a consumption shock ϵ_t. Equation (2) specifies that the growth rate of hours worked n_t depends on wage growth and consumption growth. Equation (3) specifies an autoregressive process for wage growth. The structural parameter h controls the weights on wage growth and expected future consumption growth in the consumption equation.

One cannot solve the model in (1)–(3) for the state-space form because the shock ϵ_t is added to the control equation c_t in (1). Shocks to control variables must be reparameterized as new state variables so that the model can be solved for its state-space form. We define the shocks as new state variables instead of new control variables because the shocks are exogenous. We rewrite the model in (1)–(3) so that the additive shock is a new state variable in (4)–(7). Specifically, we define a new state variable z_t such that $z_t = \epsilon_t$ and write the model as

$$c_t = (1-h)w_t + hE_tc_{t+1} + z_t \tag{4}$$
$$n_t = w_t - \gamma c_t \tag{5}$$
$$w_{t+1} = \rho w_t + \xi_{t+1} \tag{6}$$
$$z_{t+1} = \epsilon_{t+1} \tag{7}$$

That we replaced ϵ_t with the new state variable z_t is the only difference between (4) and (1). Equations (5) and (6) are identical to their counterparts, (2) and (3). Equation (7) specifies the evolution of the new state variable z_t; it is similar to (6), except that it lacks an autoregressive component.

Parameter estimation

We estimate the parameters of the model in (4)–(7) using U.S. data on consumption growth and growth in hours worked. We specify `w` and `z` as state variables. We specify `c` and `n` as control variables. Both `c` and `n` are treated as observed. We specify that `w` and `z` are subject to shocks.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (c = (1-{h})*(w) +  {h}*E(F.c) + z)
>      (n = w - {gamma}*c)
>      (F.w = {rho}*w, state)
>      (F.z = , state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -2514.2527
Iteration 1:   log likelihood =   -1264.15  (backed up)
Iteration 2:   log likelihood = -1182.8885  (backed up)
Iteration 3:   log likelihood = -1153.1143  (backed up)
Iteration 4:   log likelihood = -1152.7297  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1152.4548
Iteration 6:   log likelihood =  -1134.679
Iteration 7:   log likelihood = -1131.4427
Iteration 8:   log likelihood =  -1131.283
Iteration 9:   log likelihood = -1131.2826
Iteration 10:  log likelihood = -1131.2826
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1131.2826
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
h	.7642505	.0360272	21.21	0.000	.6936386	.8348625
gamma	.2726181	.1059468	2.57	0.010	.0649663	.4802699
rho	.6545212	.0485627	13.48	0.000	.55934	.7497023
sd(e.w)	2.904762	.1958651			2.520873	3.28865
sd(e.z)	2.077219	.1400659			1.802695	2.351743

The estimate of `h` is 0.76. Being greater than 0.5, this indicates that consumption growth depends more strongly on expected future consumption growth than on current wage growth.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4b — Including a lag of a control variable

Description Remarks and examples Also see

Description

`dsge` does not allow lags of control variables to be included in the model. The structural form of the model that is required so that the model can be solved for its state-space form does not include lags of control variables. This entry shows how to fit a DSGE model that involves a lag of control variables by defining a new state variable and rewriting the equations.

Remarks and examples

Remarks are presented under the following headings:

A model with a lagged endogenous variable

The model in (1)–(5) is similar to many in monetary economics in that it includes inertia in the interest rate because the interest rate depends on its lagged value.

$$p_t = \beta E_t p_{t+1} + \kappa y_t \tag{1}$$
$$y_t = E_t y_{t+1} - (r_t - E_t p_{t+1} - \rho_z z_t) \tag{2}$$
$$r_t = \rho_r r_{t-1} + (1-\rho_r)\left(\frac{1}{\beta}p_t + u_t\right) \tag{3}$$
$$z_{t+1} = \rho_z z_t + \epsilon_{t+1} \tag{4}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{5}$$

Equation (1) specifies the structural equation for inflation p_t. Inflation is a linear combination of expected future inflation $E_t(p_{t+1})$ and output growth y_t. Equation (2) specifies the structural equation for output growth. Output growth is a linear combination of expected future output growth $E_t(y_{t+1})$, the interest rate r_t, expected future inflation, and the state z_t. The state z_t is the first-order autoregressive process that drives output growth. Equation (3) specifies the structural equation for the interest rate. The interest rate depends on its own lagged value, the inflation rate, and the state u_t. The state u_t is the first-order autoregressive process that drives the interest rate. The control variables in this model are p_t, y_t, and r_t. The state variables are u_t and z_t.

The term involving r_{t-1} in (3) is problematic because the lag of a control variable does not fit into the structure required to solve the model for the state-space form. We accommodate this term by defining a new state variable Lr_t that equals r_{t-1} and replacing r_{t-1} in (3) with this new state variable. We define a new state instead of new control because the lagged control is predetermined, which makes it exogenous. These changes yield the model in (6)–(11).

$$p_t = \beta E_t p_{t+1} + \kappa y_t \tag{6}$$
$$y_t = E_t y_{t+1} - (r_t - E_t p_{t+1} - \rho_z z_t) \tag{7}$$
$$r_t = \rho_r Lr_t + (1-\rho_r)\left(\frac{1}{\beta}p_t + u_t\right) \tag{8}$$
$$Lr_{t+1} = r_t \tag{9}$$
$$z_{t+1} = \rho_z z_t + \epsilon_{t+1} \tag{10}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{11}$$

Parameter estimation

We will estimate the parameters in the model in (6)–(11) using data on U.S. interest rate `r` and inflation `p`.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -(r - E(f.p) - {rhoz}*z), unobserved)
>      (r = {rhor}*lr + (1-{rhor})*((1/{beta})*p + u))
>      (F.lr = r, state noshock)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -158313.11
Iteration 1:   log likelihood = -5780.6489  (backed up)
Iteration 2:   log likelihood = -1028.4602  (backed up)
Iteration 3:   log likelihood =  -1000.851  (backed up)
Iteration 4:   log likelihood =  -873.3804  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -855.55061  (not concave)
Iteration 6:   log likelihood =   -799.322  (not concave)
Iteration 7:   log likelihood = -784.44106  (not concave)
Iteration 8:   log likelihood = -776.37874  (not concave)
Iteration 9:   log likelihood =  -771.3587  (not concave)
Iteration 10:  log likelihood = -767.02524  (not concave)
Iteration 11:  log likelihood = -764.16952  (not concave)
Iteration 12:  log likelihood = -761.74111  (not concave)
Iteration 13:  log likelihood = -759.86071  (not concave)
Iteration 14:  log likelihood = -758.21296
Iteration 15:  log likelihood = -755.96039
Iteration 16:  log likelihood = -754.36723
Iteration 17:  log likelihood = -753.19361
Iteration 18:  log likelihood = -753.07315
Iteration 19:  log likelihood = -753.07026
Iteration 20:  log likelihood = -753.07026
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.07026
```

	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5026369	.0791865	6.35	0.000	.3474342	.6578397
kappa	.1760193	.0511049	3.44	0.001	.0758554	.2761831
rhoz	.9591408	.0181341	52.89	0.000	.9235986	.994683
rhor	-.0939028	.0939661	-1.00	0.318	-.278073	.0902673
rhou	.7094625	.0447807	15.84	0.000	.6216939	.7972311
sd(e.u)	2.324311	.3236236			1.69002	2.958602
sd(e.z)	.6111539	.1084981			.3985014	.8238063

Ironically, the inertia term for the interest rate `rhor` was not needed in this model. The autoregressive state u_t is probably modeling both the inertia and the persistence in the shocks to the interest rate.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4c — Including a lag of a state variable

Description Remarks and examples Also see

Description

Lags of state variables are not allowed to be included in the structural model specified in the `dsge` command. These lags do not fit into the form of a structural model that can be solved for the state-space form. However, this restriction does not prevent us from fitting models that involve the lag of a state variable. This entry shows how to define a new state variable and rewrite the equations in the required structural form.

Remarks and examples

Remarks are presented under the following headings:

A model with a lagged state variable

The model in (1)–(5) is a standard monetary model in which the state driving output growth y_t is a second-order autoregressive AR(2) process instead of a first-order autoregressive process.

$$p_t = \beta E_t(p_{t+1}) + \kappa y_t \tag{1}$$
$$y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \tag{2}$$
$$r_t = \frac{1}{\beta} p_t + u_t \tag{3}$$
$$z_{t+1} = \rho_{z1} z_t + \rho_{z2} z_{t-1} + \epsilon_{t+1} \tag{4}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{5}$$

Equation (1) specifies the structural equation for inflation p_t. Inflation is a linear combination of expected future inflation $E_t(p_{t+1})$ and output growth y_t. Equation (2) specifies the structural equation for output growth. Output growth is a linear combination of expected future output growth $E_t(y_{t+1})$, the interest rate r_t, expected future inflation, and the state z_t. The state z_t is an AR(2) process that drives output growth. Equation (3) specifies the structural equation for the interest rate. The interest rate depends on the inflation rate and the state u_t. The state u_t is an AR(1) process that drives the interest rate. The control variables in this model are p_t, y_t, and r_t. The state variables are u_t and z_t.

The AR(2) term in (4) is a problematic term because a lag of a state variable does not fit into the structure required to solve the model for the state-space form. We accommodate this term by defining a new state variable Lz_t that equals z_{t-1} and replacing z_{t-1} in (4) with this new state variable. We define a new state instead of new control because the lagged state is exogenous. These changes yield the model in (6)–(11).

$$p_t = \beta E_t(p_{t+1}) + \kappa y_t \tag{6}$$
$$y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \tag{7}$$
$$r_t = \frac{1}{\beta} p_t + u_t \tag{8}$$
$$z_{t+1} = \rho_{z1} z_t + \rho_{z2} L z_t + \epsilon_{t+1} \tag{9}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{10}$$
$$L z_{t+1} = z_t \tag{11}$$

We will estimate the parameters in the model in (6)–(11).

Parameter estimation

We estimate the parameters in the model in (6)–(11) using data on U.S. interest rate `r` and inflation `p`.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p    = {beta}*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = (1/{beta})*p + u)
>      (F.u  = {rhou}*u, state)
>      (F.z  = {rhoz1}*z + {rhoz2}*Lz, state)
>      (F.Lz = z, state noshock)
(setting technique to bfgs)
Iteration 0:   log likelihood = -9116.0985
Iteration 1:   log likelihood = -997.95396  (backed up)
Iteration 2:   log likelihood = -881.53897  (backed up)
Iteration 3:   log likelihood = -836.47478  (backed up)
Iteration 4:   log likelihood = -769.37111  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -768.80775  (not concave)
Iteration 6:   log likelihood = -760.17544  (not concave)
Iteration 7:   log likelihood = -755.99765  (not concave)
Iteration 8:   log likelihood = -754.93526  (not concave)
Iteration 9:   log likelihood = -754.42155
Iteration 10:  log likelihood =  -753.8269
Iteration 11:  log likelihood = -753.20702
Iteration 12:  log likelihood = -753.08071
Iteration 13:  log likelihood = -753.07788
Iteration 14:  log likelihood = -753.07788
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.07788
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5154874	.0770496	6.69	0.000	.364473	.6665019
kappa	.1662425	.0468473	3.55	0.000	.0744234	.2580616
rhou	.6979877	.0452071	15.44	0.000	.6093834	.7865921
rhoz1	.6735462	.2666028	2.53	0.012	.1510143	1.196078
rhoz2	.2710021	.2564596	1.06	0.291	-.2316495	.7736537
sd(e.u)	2.315262	.2992183			1.728805	2.901719
sd(e.z)	.7720678	.1716214			.4356961	1.10844

Looking at the confidence interval on `rhoz2`, we find no evidence that the AR(2) term is needed in this model.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4d — Including an expectation of a control dated by more than one period ahead

Description Remarks and examples Also see

Description

Expectations of control variables dated by more than one period ahead are not allowed in the equations specified in `dsge` because they do not appear in the required form of a structural model that can be solved for the state-space form. In this entry, we demonstrate how to fit models with these types of expectations by defining a new variable and rewriting our equations. Because the expectation dated more than one period ahead is endogenous, the new variable that we define is a new control variable.

Remarks and examples

Remarks are presented under the following headings:

The model

Equations (1)–(4) specify a model of consumption growth and growth in hours worked.

$$c_t = (1-h)w_t + hE_tc_{t+2} + r_t \tag{1}$$
$$n_t = w_t - \gamma c_t \tag{2}$$
$$w_{t+1} = \rho_w w_t + \xi_{t+1} \tag{3}$$
$$r_{t+1} = \rho_r r_t + \epsilon_{t+1} \tag{4}$$

Equation (1) specifies that consumption growth c_t is a linear combination of wage growth w_t, the expected value of consumption growth two periods ahead E_tc_{t+2}, and the interest rate r_t. Equation (2) specifies that the growth rate of hours worked n_t depends on wage growth and consumption growth. Equations (3) and (4) specify a first-order autoregressive process for wage growth and for the interest rate, respectively. The control variables are c_t and n_t, and the state variables are w_t and r_t.

The expected value of consumption growth two periods ahead in (1) is a problematic term because it does not fit into the structure required to solve for the state-space form. The structure requires that expectations be only of one-period ahead values. We accommodate this term by defining a new control variable Fc_t that equals $E_t(c_{t+1})$ and replacing c_{t+1} in (1) with this new control variable. We define a new control variable instead of a new state variable because the expected future consumption is endogenous instead of exogenous. These changes yield the model in (5)–(9).

$$c_t = (1-h)w_t + hE_t(Fc_{t+1}) + r_t \tag{5}$$
$$n_t = w_t - \gamma c_t \tag{6}$$
$$Fc_t = E_t(c_{t+1}) \tag{7}$$
$$w_{t+1} = \rho_w w_t + \xi_{t+1} \tag{8}$$
$$r_{t+1} = \rho_r r_t + \epsilon_{t+1} \tag{9}$$

The new (7) defines the new control variable Fc_t as the expected value of next period's consumption. Equation (5) is (1) but with c_{t+2} replaced with the new control Fc_{t+1}.

There are now three control variables and only two shocks, so we treat the new control variable Fc_t as unobserved. There is a logic to this process. The one-step ahead structure of state-space models makes it impossible to solve for terms like $E_t(c_{t+2})$. In contrast, it is possible to solve for terms like $E_t\{E_t(c_{t+1})\}$ as long as the unobserved $E_t(c_{t+1})$ is determined by another equation, which in this case is (7). In fact, this example illustrates a part of the recursive dynamic modeling approach at the heart of the DSGE approach to macroeconomics.

Parameter estimation

We estimate the parameters of the model in (5)–(9) using U.S. data on consumption growth and growth in hours worked.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (c   = (1-{h})*(w) + {h}*E(F.fc) + r)
>      (n   = w - {gamma}*c)
>      (fc  = E(F.c), unobserved)
>      (F.w = {rho_w}*w, state)
>      (F.r = {rho_r}*r, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -2423.7325
Iteration 1:   log likelihood = -1284.9295  (backed up)
Iteration 2:   log likelihood = -1193.2234  (backed up)
Iteration 3:   log likelihood = -1180.1787  (backed up)
Iteration 4:   log likelihood = -1175.3563  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1171.0676  (backed up)
Iteration 6:   log likelihood = -1152.0355
Iteration 7:   log likelihood = -1134.7089
Iteration 8:   log likelihood = -1130.0791
Iteration 9:   log likelihood = -1129.9372
Iteration 10:  log likelihood = -1129.9357
Iteration 11:  log likelihood = -1129.9357
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1129.9357
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
h	.661791	.0427917	15.47	0.000	.5779208	.7456611
gamma	.273381	.1120734	2.44	0.015	.0537212	.4930408
rho_w	.6545228	.0485628	13.48	0.000	.5593415	.7497041
rho_r	.1050387	.0638171	1.65	0.100	-.0200406	.230118
sd(e.w)	2.905807	.2023204			2.509267	3.302348
sd(e.r)	2.049915	.1437862			1.7681	2.331731

The estimate of h is greater than 0.5, so we conclude that slightly more of the unobserved wage state is allocated to the expected growth in consumption two periods ahead than to current consumption.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4e — Including a second-order lag of a control

Description Remarks and examples Also see

Description

Some DSGE models capture delayed effects by including a second-order lag of a control variable and excluding the first-order lag. The second-order lag is a problematic term that does not fit into the form required to solve a structural model for its state-space form. This entry shows how to solve this problem by defining new state variables and rewriting the equations.

Remarks and examples

Remarks are presented under the following headings:

The model

Consider a model in which changes in hours worked take two periods to adjust because next period's hours have already been budgeted. In this model, the second-order lag of changes in hours worked is included and the first-order lag is excluded. Equations (1)–(4) specify such a model of growth in hours worked and of consumption growth.

$$n_t = b_1 n_{t-2} + w_t - \gamma c_t \tag{1}$$
$$c_t = (1-h) w_t + h E_t c_{t+1} + r_t \tag{2}$$
$$w_{t+1} = \rho w_t + \xi_{t+1} \tag{3}$$
$$r_{t+1} = \epsilon_{t+1} \tag{4}$$

Equation (1) specifies that the growth rate of hours worked n_t depends on a second-order lag of itself, wage growth w_t, and consumption growth c_t. Equation (2) specifies that consumption growth is a linear combination of wage growth, expected future consumption growth $E_t c_{t+1}$, and the interest rate r_t. Equation (3) specifies an autoregressive process for wage growth. Equation (4) specifies that interest rate is just a shock. The control variables are n_t and c_t. The state variables are w_t and r_t.

One cannot solve the model in (1)–(4) for the state-space form because the problematic term $b_1 n_{t-2}$ does not fit into the required form. To accommodate this term, we define two new state variables, one for n_{t-1} and one for n_{t-2}. We define new state variables instead of new control variables because lags of the control are predetermined and thus exogenous. The model with new state variables is

$$n_t = b_1 L2n_t + w_t - \gamma c_t \tag{5}$$
$$c_t = (1-h)w_t + hE_t c_{t+1} + r_t \tag{6}$$
$$w_{t+1} = \rho w_t + \xi_{t+1} \tag{7}$$
$$r_{t+1} = \epsilon_{t+1} \tag{8}$$
$$Ln_{t+1} = n_t \tag{9}$$
$$L2n_{t+1} = Ln_t \tag{10}$$

Equation (9) defines the new state for n_{t-1}, and (10) defines $L2n_t$ to be the new state for n_{t-2}. The $L2n_t$ in (5) replaces n_{t-2} in (1).

Parameter estimation

We specify `n` and `t` as observed control equations. We specify `w`, `r`, `Ln`, and `L2n` as state equations. We specify that `w` and `r` are subject to shocks; the new states to accommodate n_{t-2} are not subject to shocks.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (n = {b1}*L2n + w - {gamma}*c)
>       (c = (1-{h})*w + {h}*E(F.c) + r)
>       (F.w = {rho}*w, state)
>       (F.r = , state)
>       (F.L2n = Ln, state noshock)
>       (F.Ln = n, state noshock)
(setting technique to bfgs)
Iteration 0:   log likelihood = -2325.1996
Iteration 1:   log likelihood = -1277.0146  (backed up)
Iteration 2:   log likelihood = -1193.4512  (backed up)
Iteration 3:   log likelihood = -1189.3181  (backed up)
Iteration 4:   log likelihood = -1188.2629  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1187.9872  (backed up)
Iteration 6:   log likelihood = -1147.3379
Iteration 7:   log likelihood = -1131.5153
Iteration 8:   log likelihood = -1129.0358
Iteration 9:   log likelihood = -1129.0181
Iteration 10:  log likelihood = -1129.0181
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1129.0181
```

	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
b1	.1320845	.0608727	2.17	0.030	.0127763	.2513927
gamma	.360925	.1298387	2.78	0.005	.1064457	.6154042
h	.7238121	.0406724	17.80	0.000	.6440957	.8035285
rho	.6177969	.0533568	11.58	0.000	.5132195	.7223743
sd(e.w)	3.033799	.2423856			2.558732	3.508866
sd(e.r)	1.970288	.1574527			1.661687	2.27889

Looking at the confidence interval for `b1`, we conclude that the second-order lag of hours growth impacts current hours growth.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4f — Including an observed exogenous variable

Description Remarks and examples Also see

Description

All the observed variables in a DSGE model must be modeled as endogenous control variables. This requirement implies that there is no reduced form for the endogenous variables as a function of observed exogenous variables. Any variable that is theoretically exogenous must be modeled.

Mechanically, the solution is to define a control variable that is equal to a state variable that models the exogenous process. We clarify this issue by discussing a model in which the price of oil is theoretically exogenous.

Remarks and examples

Remarks are presented under the following headings:

The model

We model an economy in which the growth rate of the trade-weighted exchange rate is exogenous and in which it affects inflation. Henceforth, we call the trade-weighted exchange rate just the exchange rate. We begin by adding the growth rate of the exchange rate to the inflation equation in the New Keynesian model of [DSGE] **intro 1**.

$$x_t = E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \tag{1}$$

$$r_t = \frac{1}{\beta}p_t + u_t \tag{2}$$

$$p_t = \beta E_t(p_{t+1}) + \kappa x_t + \psi e_t \tag{3}$$

$$u_{t+1} = \rho_u u_t + \epsilon_{t+1} \tag{4}$$

$$g_{t+1} = \rho_g g_t + \xi_{t+1} \tag{5}$$

x_t is the output gap, which is modeled as an unobserved control variable. r_t is the interest rate, which is modeled as an observed control variable. p_t is the inflation rate, which is modeled as an observed control variable. g_t and u_t are first-order autoregressive state variables. e_t is the growth rate of the exchange rate, which we want to model as an observed exogenous variable.

The model in (1)–(5) cannot be solved because there is no equation for how the observed e_t evolves over time. A first-order autoregressive process [AR(1)] is a standard approximation to an exogenous variable, like the growth in exchange rate. We complete the model by adding an AR(1) for the unobserved state variable es_t and an equation linking the unobserved es_t to the observed e_t.

$$x_t = E_t(x_{t+1}) - \{r_t - E_t(p_{t+1}) - g_t\} \tag{6}$$
$$r_t = \frac{1}{\beta}p_t + u_t \tag{7}$$
$$p_t = \beta E_t(p_{t+1}) + \kappa x_t + \psi es_t \tag{8}$$
$$e_t = es_t \tag{8a}$$
$$u_{t+1} = \rho_u u_t + \epsilon_{t+1} \tag{9}$$
$$g_{t+1} = \rho_g g_t + \xi_{t+1} \tag{10}$$
$$es_{t+1} = \rho_e es_t + \eta_{t+1} \tag{11}$$

New (8a) specifies how the unobserved state es_t is transformed into the observed control variable e_t. Equation (11) specifies that the unobserved state es_t is an AR(1) process. Other equations are unchanged.

Parameter estimation

We fit this model using data on the U.S. interest rate `r`, inflation rate `p`, and the growth rate of the exchange rate `e`. The equation `(e = es)` links the observed variable `e` to the unobserved state variable `es`.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (x = E(F.x) -(r - E(f.p) - g), unobserved)
>      (r = 1/{beta}*p + u)
>      (p = {beta}*E(F.p) + {kappa}*x + {psi}*es)
>      (e = es)
>      (F.u = {rho_u}*u, state)
>      (F.g = {rho_g}*g, state)
>      (F.es = {rho_e}*es, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -24249.537
Iteration 1:   log likelihood = -8517.9588  (backed up)
Iteration 2:   log likelihood = -1808.8075  (backed up)
Iteration 3:   log likelihood = -1625.0334  (backed up)
Iteration 4:   log likelihood = -1600.0843  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1557.8925  (not concave)
Iteration 6:   log likelihood =  -1512.705  (not concave)
Iteration 7:   log likelihood = -1481.8476  (not concave)
Iteration 8:   log likelihood = -1278.0033  (not concave)
Iteration 9:   log likelihood = -1198.5095  (not concave)
Iteration 10:  log likelihood = -1186.0134
Iteration 11:  log likelihood = -1177.2355  (not concave)
Iteration 12:  log likelihood = -1175.3012
Iteration 13:  log likelihood = -1174.3886
Iteration 14:  log likelihood = -1172.7244
Iteration 15:  log likelihood = -1172.3166
Iteration 16:  log likelihood = -1172.0638
Iteration 17:  log likelihood = -1172.0376
Iteration 18:  log likelihood = -1172.0364
Iteration 19:  log likelihood = -1172.0364
```

```
DSGE model
Sample: 1973q2 - 2015q4                         Number of obs     =        171
Log likelihood = -1172.0364
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5106735	.1022178	5.00	0.000	.3103303	.7110167
kappa	.0901563	.0334284	2.70	0.007	.0246378	.1556747
psi	.01373	.0037684	3.64	0.000	.006344	.021116
rho_u	.7789341	.047395	16.43	0.000	.6860415	.8718266
rho_g	.9597842	.0206193	46.55	0.000	.9193712	1.000197
rho_e	.2372214	.0742225	3.20	0.001	.0917479	.3826948
sd(e.u)	2.206087	.3466166			1.526731	2.885443
sd(e.g)	.703933	.1545767			.4009683	1.006898
sd(e.es)	10.48932	.5671986			9.37763	11.60101

From the estimates for `psi`, it appears that the growth of the exchange rate impacts inflation.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 4g — Correlated state variables

Description Remarks and examples Also see

Description

Many models include correlated state variables. We illustrate how to specify correlated state variables in a model of output growth y_t and inflation p_t.

Remarks and examples

Remarks are presented under the following headings:

The model

We model output growth y_t and inflation p_t as functions of domestic and international factors. The domestic factor g_t that drives output growth is a first-order autoregressive process that is also affected by the international factor z_t. The international factor that drives inflation is a simple first-order autoregressive process.

Mathematically, the model is

$$y_t = E_t y_{t+1} + \alpha p_t + g_t \tag{1}$$
$$p_t = z_t \tag{2}$$
$$g_{t+1} = \rho_g g_t + \rho_{gz} z_t + \xi_{t+1} \tag{3}$$
$$z_{t+1} = \rho_z z_t + \epsilon_{t+1} \tag{4}$$

Equation (1) specifies that output growth depends on expected future output growth $E_t y_{t+1}$, inflation p_t, and the domestic factor g_t. This equation has the form of an aggregate demand curve, and the parameter α is referred to as the slope of the aggregate demand curve. It should be negative. Equation (2) specifies that inflation is entirely driven by the international factor z_t. Equations (3) and (4) specify that the factors g_t and z_t follow a first-order vector autoregressive process with parameters ρ_g, ρ_{gz} and ρ_z and with shocks ξ_{t+1} and ϵ_{t+1}. The factors z_t and g_t are the state variables, and p_t and y_t are the observed control variables.

Parameter estimation

From the U.S. macroeconomic data, we use the GDP-growth data in y and the inflation data in p and estimate the parameters of this model.

```
. dsge (y = E(F.y) + {alpha}*p + g)
>      (p = z)
>      (F.g = {rho_g}*g + {rho_gz}*z, state)
>      (F.z = {rho_z}*z, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -2106.7245
Iteration 1:   log likelihood = -1563.7386  (backed up)
Iteration 2:   log likelihood = -1363.6417  (backed up)
Iteration 3:   log likelihood = -1289.3079  (backed up)
Iteration 4:   log likelihood = -1274.5732  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1175.4471  (not concave)
Iteration 6:   log likelihood = -1121.5017  (not concave)
Iteration 7:   log likelihood = -1111.6839  (not concave)
Iteration 8:   log likelihood = -1104.5102  (not concave)
Iteration 9:   log likelihood = -1098.5517  (not concave)
Iteration 10:  log likelihood = -1085.8018  (not concave)
Iteration 11:  log likelihood =  -1074.273  (not concave)
Iteration 12:  log likelihood = -1071.4913  (not concave)
Iteration 13:  log likelihood = -1069.9378  (not concave)
Iteration 14:  log likelihood = -1060.2549  (not concave)
Iteration 15:  log likelihood =  -1057.046
Iteration 16:  log likelihood = -1040.5911  (not concave)
Iteration 17:  log likelihood = -1028.4304  (not concave)
Iteration 18:  log likelihood = -1023.5103  (not concave)
Iteration 19:  log likelihood = -1021.3008
Iteration 20:  log likelihood = -1018.6421  (not concave)
Iteration 21:  log likelihood = -1017.6527
Iteration 22:  log likelihood =  -1017.315
Iteration 23:  log likelihood =  -1017.164
Iteration 24:  log likelihood = -1017.1592
Iteration 25:  log likelihood = -1017.1592
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1017.1592
```

	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
alpha	-.1130023	.1554398	-0.73	0.467	-.4176587	.1916541
rho_g	.335777	.0610763	5.50	0.000	.2160696	.4554843
rho_gz	.0443504	.09013	0.49	0.623	-.1323013	.221002
rho_z	.8626566	.0319007	27.04	0.000	.8001324	.9251807
sd(e.g)	2.184805	.2241105			1.745557	2.624054
sd(e.z)	1.146947	.0519234			1.045179	1.248715

The slope of the aggregate demand curve, α, is estimated to be negative as we expected, but the confidence interval is wide and includes zero. The imprecision in `rho_gz` has caused imprecision in the estimate of α.

Also see

[DSGE] **intro 2** — Learning the syntax

[DSGE] **intro 4** — Writing a DSGE in a solvable form

Title

intro 5 — Stability conditions

Description Remarks and examples References Also see

Description

We can estimate only the parameters of models that we can solve, and we can solve only models that satisfy a stability condition. This entry discusses this condition, how it affects estimation in practice, and how to find initial values that satisfy it.

Remarks and examples

Remarks are presented under the following headings:

Why we care about stability

DSGE models are dynamic systems subject to random shocks. DSGE models that do not spiral out of control or converge to a single point when shocked are said to be "saddle-path stable". We can solve only saddle-path stable DSGE models, and we can estimate only the parameters of models we can solve.

The parameter values determine whether a DSGE model is saddle-path stable. Most DSGE models are saddle-path stable for some parameter values and not for other values. As we discussed in *How to write down a DSGE* in [DSGE] **intro 1**, the structural form of a DSGE model is

$$\mathbf{A}_0\mathbf{y}_t = \mathbf{A}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{A}_2\mathbf{y}_t + \mathbf{A}_3\mathbf{x}_t \tag{1}$$

$$\mathbf{B}_0\mathbf{x}_{t+1} = \mathbf{B}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{B}_2\mathbf{y}_t + \mathbf{B}_3\mathbf{x}_t + \mathbf{C}\boldsymbol{\epsilon}_{t+1} \tag{2}$$

Given this structural form, the values in the $\mathbf{A}_i$ and $\mathbf{B}_j$ matrices determine saddle-path stability.

In the process of solving the structural form in (1) and (2) for the state space, the Klein (2000) solver computes the generalized eigenvalues of a matrix formed from the values in the $\mathbf{A}_i$ and $\mathbf{B}_j$ matrices. An eigenvalue is said to be stable when its absolute value is less than 1. The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model.

Once the maximization algorithm gets started, saddle-path stability usually is not an issue. Problems with saddle-path stability usually occur when trying to get the maximization process started. The initial values for the maximization algorithm must imply saddle-path stability because we cannot solve a DSGE model for parameter values that do not imply saddle-path stability. When the default initial values do not imply saddle-path stability, `dsge` searches for saddle-path-stable parameters, but there is no guarantee that it will find any.

`dsge` uses a series of small positive numbers as the default initial values for the structural parameters that appear in the $\mathbf{A}$ and $\mathbf{B}$ matrices. If you know that your model is not saddle-path stable when a parameter, say, β, is set to 0.1, you should use the `from()` option to specify an initial value that does imply saddle-path stability. Alternatively, you could reparameterize the model so that $\beta = 0.1$ does imply saddle-path stability, but this solution tends to be more work than changing the initial value.

In rare cases, when the true parameters are close to values that are not saddle-path stable, the maximum likelihood estimator will suffer from convergence problems. Better initial values can help find the solution in these cases.

What if the initial values are not saddle-path stable?

If the initial values do not imply a saddle-path stable solution and `dsge` cannot find any such values, you will see

```
. dsge ...
model is not saddle-path stable at current parameter values
    The number of stable eigenvalues is greater than the number of state
    variables.
  (output omitted)
```

or

```
. dsge ...
model is not saddle-path stable at current parameter values
    The number of stable eigenvalues is less than the number of state
    variables.
  (output omitted)
```

In either case, you should specify the `solve` and `noidencheck` options and subsequently use `estat stable` to find the problem. `solve` specifies that `dsge` only solve the model and not proceed with estimation, and `noidencheck` suppresses the check of whether the parameters are identified. `estat stable` will display the eigenvalues so that you can determine which ones are saddle-path stable and which ones are not. The trick is to change the starting values so that the number of stable eigenvalues equals the number of states in the model. Once you have determined which parameters are problematic, you can use the `from()` option to change the initial values to ones that do not have this problem.

Textbooks like those by Canova (2007), DeJong and Dave (2011), Ljungqvist and Sargent (2012), and Woodford (2003) are full of examples in which some structural parameters must be either greater than 1 or less than 1 for the model to be saddle-path stable. Similarly, some parameters must be either greater than or less than 0 for the model to be saddle-path stable. Moving one structural parameter over one of these boundaries and looking at how the change affects the number of stable eigenvalues can help find a vector of parameters that yields a saddle-path stable solution.

That the model that you are trying to fit is likely related to other models in the literature or in the textbooks suggests another strategy. Any such references probably contain some discussion of the set of values that yield saddle-path stable solutions.

In especially difficult cases, you can use the strategy of solving the problem for a smaller model and building back up. You temporarily remove equations from the model, find a parameter vector that yields a saddle-path stable solution for the smaller model, and then progressively add back equations as you find initial values that yield saddle-path stable solutions.

▷ Example 1: Finding saddle-path stable initial values

Equations (1)–(6) model the observed control variable (inflation) p_t, the unobserved control variable (output gap) y_t, and the observed control variable (interest rate) r_t as functions of the states u_t and z_t. Lz_{t+1} is an auxiliary state that allows z_t to be a second-order process instead of a first-order process.

$$p_t = (1/\gamma)E_t(p_{t+1}) + \kappa y_t \tag{3}$$
$$y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \tag{4}$$
$$r_t = \gamma p_t + u_t \tag{5}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{6}$$
$$z_{t+1} = \rho_{z1} z_t + \rho_{z2} L z_t + \epsilon_{t+1} \tag{7}$$
$$Lz_{t+1} = z_t \tag{8}$$

We try to fit this model using default starting values below.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p    = (1/{gamma})*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = {gamma}*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock)
model is not saddle-path stable at current parameter values
    The number of stable eigenvalues is greater than the number of state
    variables.
r(498);
```

The default initial values specify a model that is not saddle-path stable. We specify options `solve` and `noidencheck`, which cause the command to attempt only to solve the model and to skip the identification check.

```
. dsge (p    = (1/{gamma})*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = {gamma}*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock),
>      solve noidencheck
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood =          .
```

	OIM				
	Coef.	Std. Err.	z	P>|z|	[95% Conf. Interval]
/structural					
gamma	.21	.	.	.	. .
kappa	.22	.	.	.	. .
rho_u	.23	.	.	.	. .
rho_z1	.24	.	.	.	. .
rho_z2	.25	.	.	.	. .
sd(e.u)	1	.			. .
sd(e.z)	1	.			. .

```
Note: Skipped identification check.
Warning: Model cannot be solved at current parameter values. Current values
         imply a model that is not saddle-path stable.
```

`dsge` displays the initial values used in the attempted solution, and it displays a warning message indicating that the model cannot be solved at these values because they imply a model that is not saddle-path stable.

We use `estat stable` to look at the eigenvalues implied by the initial values.

```
. estat stable
Stability results
```

	Eigenvalues
stable	-.3942
stable	.6342
stable	.21
stable	.23
unstable	2.605e+16
unstable	1.046

```
The process is not saddle-path stable.
The process is saddle-path stable when there are 3 stable eigenvalues and 3
    unstable eigenvalues.
```

The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model. The initial values yield four stable eigenvalues, but there are only three states in the model. So the model is not saddle-path stable at these values.

The initial value for `gamma` in the output from the failed solution attempt was 0.21. From (3), we see that the coefficient on expected future inflation is greater than one when `gamma` is less than one. We suspect that this relationship could be causing the lack of saddle-path stability. Below we use option `from()` to specify 1.2 as an initial value for `gamma` and attempt to solve the model.

```
. dsge (p    = (1/{gamma})*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = {gamma}*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock),
>      solve noidencheck from(gamma=1.2)
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -1504.7564
```

	OIM				
	Coef.	Std. Err.	z	P>|z|	[95% Conf. Interval]
/structural					
gamma	1.2	.	.	.	. .
kappa	.22	.	.	.	. .
rho_u	.23	.	.	.	. .
rho_z1	.24	.	.	.	. .
rho_z2	.25	.	.	.	. .
sd(e.u)	1	.			. .
sd(e.z)	1	.			. .

```
Note: Skipped identification check.
Note: Model solved at specified parameters.
```

There is no warning message, so the parameter values yielded a saddle-path stable solution. As a final illustration of this point, we display the eigenvalues below.

```
. estat stable

Stability results
```

	Eigenvalues
stable	-.3942
stable	.6342
stable	.23
unstable	5.380e+16
unstable	1.2
unstable	1.264

```
The process is saddle-path stable.
```

We could now proceed with estimation.

◁

References

Canova, F. 2007. *Methods for Applied Macroeconomic Research*. Princeton, NJ: Princeton University Press.

DeJong, D. N., and C. Dave. 2011. *Structural Macroeconometrics*. 2nd ed. Princeton, NJ: Princeton University Press.

Klein, P. 2000. Using the generalized Schur form to solve a multivariate linear rational expectations model. *Journal of Economic Dynamics and Control* 24: 1405–1423.

Ljungqvist, L., and T. J. Sargent. 2012. *Recursive Macroeconomic Theory*. 3rd ed. Cambridge, MA: MIT Press.

Woodford, M. 2003. *Interest and Prices: Foundations of a Theory of Monetary Policy*. Princeton, NJ: Princeton University Press.

Also see

[DSGE] **intro 6** — Identification

[DSGE] **intro 7** — Convergence problems

[DSGE] **estat stable** — Check stability of system

Title

intro 6 — Identification

Description Remarks and examples Reference Also see

Description

We can estimate only the parameters of DSGE models whose parameters are identified. We discuss how to find the identification problems and how to fix them.

Remarks and examples

The parameters that maximize the likelihood function are the maximum likelihood (ML) estimates of ML estimators like the one in `dsge`. When there is a unique vector of parameters that maximizes the likelihood function, the parameters are identified and we can estimate them. When there is more than one parameter vector that produces the same maximum value of the likelihood function, the parameters are not identified, and we cannot estimate them because there is no way to choose among the sets of equally best parameters.

`dsge` and all other commands that implement ML estimators check for identification at the solution by checking that the matrix of second derivatives, known as the Hessian, is full rank at the maximum. This test works only when the maximization algorithm arrives at a candidate maximum. In addition, it provides no information about which parameters are not identified.

For many models, when the parameters are not identified, the only output produced by a standard ML estimator is an ever growing series of "not concave" messages. This issue is especially relevant to DSGE models because it is easy to write down theoretical DSGE models whose parameters are not identified.

`dsge` uses the Iskrev (2010) diagnostic to resolve this issue. The Iskrev diagnostic detects lack of parameter identification and reports which parameters are not identified at specific parameter values. By default, `dsge` performs the Iskrev diagnostic at the initial values and at the solution values.

Instead of looking at the rank of the Hessian, the Iskrev (2010) diagnostic checks that all the parameters affect the autocovariances of the observed control variables. When there is a one-to-one relationship between the autocovariances for the observed control variables and the model parameters, the parameters are identified. Intuitively, this one-to-one relationship allows us to back out the model parameters from the autocovariances that we can always estimate from the data. When some of the parameters do not affect the autocovariances of the observed control variables, there is no one-to-one relationship and the model parameters are not identified.

Simply adding restrictions to parameters can make the parameters of an unidentified model identified. This fact is frustrating but intuitive. When the parameters are not identified, they are not uniquely determined by the data generated by the true model. Only adding restrictions, either by defining constraints or by changing the equations, can identify the parameters.

▷ Example 1: A model with two unidentified parameters

Consider the following model of inflation p_t, output growth y_t, and the interest rate r_t.

$$
\begin{aligned}
p_t &= \beta E_t(p_{t+1}) + \kappa y_t \\
y_t &= E_t(y_{t+1}) - \gamma\{r_t - E_t(p_{t+1}) - \rho z_t\} \\
\beta r_t &= p_t + \beta u_t \\
z_{t+1} &= \rho z_t + \epsilon_{t+1} \\
u_{t+1} &= \delta u_t + \xi_{t+1}
\end{aligned}
$$

This model specifies how the observed control variables p_t and r_t depend on the unobserved control variable y_t, on the state variables z_t and u_t, and on the shocks ϵ_t and ξ_t.

Let's try to estimate the parameters of this model.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -{gamma}*(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state)
identification failure at starting values
    Constrain some parameters or specify option noidencheck.  Likely source of
    identification failure: {kappa} {gamma}
r(498);
```

The error message indicates that `kappa` and `gamma` do not influence the autocovariances independently; they are linearly dependent. If a list of linearly dependent parameters is supplied, then you must constrain all but one of them prior to estimation. We constrain `gamma`.

```
. constraint 2 _b[gamma]=1
```

And we apply the constraint to the model:

```
. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -{gamma}*(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state),
>      constraint(2)
(setting technique to bfgs)
Iteration 0:   log likelihood =  -128443.1
Iteration 1:   log likelihood = -3610.8865  (backed up)
Iteration 2:   log likelihood = -1066.3053  (backed up)
Iteration 3:   log likelihood =  -919.1875  (backed up)
Iteration 4:   log likelihood = -815.49521  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -805.28492  (backed up)
Iteration 6:   log likelihood = -790.04311
Iteration 7:   log likelihood = -773.86016
Iteration 8:   log likelihood =  -755.6535
Iteration 9:   log likelihood = -753.96757
Iteration 10:  log likelihood = -753.57754
Iteration 11:  log likelihood = -753.57134
Iteration 12:  log likelihood = -753.57131
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
 ( 1)  [/structural]gamma = 1
```

	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5146687	.078349	6.57	0.000	.3611076	.6682298
kappa	.1659038	.0474064	3.50	0.000	.0729889	.2588187
gamma	1	(constrained)				
rhoz	.9545255	.0186424	51.20	0.000	.9179871	.9910639
rhou	.7005496	.0452601	15.48	0.000	.6118414	.7892577
sd(e.u)	2.318197	.3047421			1.720914	2.915481
sd(e.z)	.650711	.1123848			.4304409	.8709811

The remaining free parameters of the model are identified and can be estimated.

◁

Keep in mind that the Iskrev (2010) diagnostic is performed at specific parameter values. Sometimes, the parameters of a model cannot be identified at a vector of parameter values but can be identified at practically any other nearby vector. This problem arises when simple initial values cause terms to cancel out. If you suspect that the identification diagnostic is finding a problem because of initial values, you can either specify `noidencheck` or specify other seemingly random initial values.

▷ Example 2: Suppressing the identification check

We suppress the identification check in an attempt to estimate the parameters without imposing additional constraints. We specify the `iterate(50)` option in case the parameters are not identified.

```
. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -{gamma}*(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state),
>      noidencheck iterate(50)
(setting technique to bfgs)
Iteration 0:   log likelihood = -1597221.6
Iteration 1:   log likelihood =  -5097.544  (backed up)
Iteration 2:   log likelihood = -1073.3293  (backed up)
Iteration 3:   log likelihood = -978.36896  (backed up)
Iteration 4:   log likelihood = -818.00268  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -808.99197  (not concave)
Iteration 6:   log likelihood = -778.94324  (not concave)
Iteration 7:   log likelihood = -761.39371  (not concave)
Iteration 8:   log likelihood = -755.33513  (not concave)
Iteration 9:   log likelihood = -754.55068
Iteration 10:  log likelihood = -753.60217  (not concave)
Iteration 11:  log likelihood =  -753.5846  (not concave)
Iteration 12:  log likelihood = -753.58018  (not concave)
Iteration 13:  log likelihood = -753.57404  (not concave)
Iteration 14:  log likelihood = -753.57267  (not concave)
Iteration 15:  log likelihood =  -753.5721
Iteration 16:  log likelihood = -753.57131
Iteration 17:  log likelihood = -753.57131  (not concave)
Iteration 18:  log likelihood = -753.57131  (not concave)
```

```
Iteration 19:  log likelihood = -753.57131  (not concave)
Iteration 20:  log likelihood = -753.57131  (not concave)
Iteration 21:  log likelihood = -753.57131  (not concave)
Iteration 22:  log likelihood = -753.57131  (not concave)
Iteration 23:  log likelihood = -753.57131  (not concave)
Iteration 24:  log likelihood = -753.57131  (not concave)
Iteration 25:  log likelihood = -753.57131  (not concave)
Iteration 26:  log likelihood = -753.57131  (not concave)
Iteration 27:  log likelihood = -753.57131  (not concave)
Iteration 28:  log likelihood = -753.57131  (not concave)
Iteration 29:  log likelihood = -753.57131  (not concave)
Iteration 30:  log likelihood = -753.57131  (not concave)
Iteration 31:  log likelihood = -753.57131  (not concave)
Iteration 32:  log likelihood = -753.57131  (not concave)
Iteration 33:  log likelihood = -753.57131  (not concave)
Iteration 34:  log likelihood = -753.57131  (not concave)
Iteration 35:  log likelihood = -753.57131  (not concave)
Iteration 36:  log likelihood = -753.57131  (not concave)
Iteration 37:  log likelihood = -753.57131  (not concave)
Iteration 38:  log likelihood = -753.57131  (not concave)
Iteration 39:  log likelihood = -753.57131  (not concave)
Iteration 40:  log likelihood = -753.57131  (not concave)
Iteration 41:  log likelihood = -753.57131  (not concave)
Iteration 42:  log likelihood = -753.57131  (not concave)
Iteration 43:  log likelihood = -753.57131  (not concave)
Iteration 44:  log likelihood = -753.57131  (not concave)
Iteration 45:  log likelihood = -753.57131  (not concave)
Iteration 46:  log likelihood = -753.57131  (not concave)
Iteration 47:  log likelihood = -753.57131  (not concave)
Iteration 48:  log likelihood = -753.57131  (not concave)
Iteration 49:  log likelihood = -753.57131  (not concave)
Iteration 50:  log likelihood = -753.57131  (not concave)
convergence not achieved
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5146672	.078349	6.57	0.000	.3611059	.6682285
kappa	.424358	.	.	.	.	.
gamma	.3909567	.1117155	3.50	0.000	.1719983	.609915
rhoz	.9545256	.0186424	51.20	0.000	.9179872	.991064
rhou	.7005485	.0452604	15.48	0.000	.6118397	.7892573
sd(e.u)	2.318203	.3047446			1.720915	2.915492
sd(e.z)	.6507122	.1123847			.4304422	.8709822

```
Note: Parameters not identified at reported values.
```

We see a long list of `not concave` messages, indicating that the parameters are indeed not identified without additional constraints. We could remove the option limiting the number of iterations, and the command should produce `not concave` iterations indefinitely.

◁

Reference

Iskrev, N. 2010. Local identification in DSGE models. *Journal of Monetary Economics* 57: 189–202.

Also see

[DSGE] **intro 5** — Stability conditions

[DSGE] **intro 7** — Convergence problems

Title

intro 7 — Convergence problems

Description Remarks and examples Also see

Description

Fitting DSGE models is notoriously difficult. We discuss the causes of convergence problems and some solutions.

Remarks and examples

When the iteration process never gets started or when it never ends are two types of convergence problems. When it never gets started, you will see something like

```
. dsge ...
model could not be solved at starting values
    equilibrium is locally indeterminate
  (output omitted)
```

This error occurs when the initial values specify a model that violates the stability conditions required for DSGE estimation; see [DSGE] **intro 5** for a discussion of this issue and some solutions.

When the iteration process never ends, the cause is either that there is no unique solution or that the optimizer cannot find the unique solution. When there is no unique solution, the parameters are said to be not identified. In this case, you see `(not concave)` after each iteration, like

```
. dsge ...
  (output omitted)
Iteration 50: log likelihood = -337504.44 (not concave)
Iteration 51: log likelihood = -337503.52 (not concave)
Iteration 52: log likelihood = -337502.13 (not concave)
.
.
.
```

See [DSGE] **intro 6** for a discussion of this issue and some solutions.

Even when the iterations get started and the parameters are identified, convergence problems are still common. In these cases, there are usually many parameters.

Changing the optimization technique is the simplest method to overcome a convergence problem, and it can be surprisingly effective. By default, `dsge` uses the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm for five iterations before switching to a modified Newton–Raphson (NR) algorithm. The BFGS algorithm is especially effective at finding a maximum of the DSGE log-likelihood function from poor initial values. In some cases, simply specifying `technique(bfgs 200 nr)` can overcome the convergence problem.

The best way to overcome a convergence problem is to provide better initial values. The optimizer used by `dsge` will almost always converge when it is provided with good initial values and the parameters are identified. Specifying good initial values that come from theory or previous empirical work can solve your convergence problem. You can specify initial values inside the scalar substitutable expressions or by using the `from()` option; see examples 1 and 2 for details.

Sometimes, you have a convergence problem and do not have good initial values. One way to get initial values is to estimate a series of progressively less constrained models. This process usually produces convergence when the parameters are identified. See example 3 for details.

▷ Example 1: Specifying starting values in scalar substitutable expressions

Models with lagged state variables and other models that allow for more persistent processes can exhibit convergence problems. Equations (1)–(6) model the observed control variable (inflation) p_t, the unobserved control variable (output gap) y_t, and the observed control variable (interest rate) r_t as functions of the states u_t and z_t. Lz_{t+1} is an auxiliary state that allows z_t to be a second-order process instead of a first-order process.

$$p_t = \beta E_t(p_{t+1}) + \kappa y_t \tag{1}$$
$$y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - z_t\} \tag{2}$$
$$r_t = \frac{1}{\beta} p_t + u_t \tag{3}$$
$$u_{t+1} = \rho_u u_t + \xi_{t+1} \tag{4}$$
$$z_{t+1} = \rho_{z1} z_t + \rho_{z2} L z_t + \epsilon_{t+1} \tag{5}$$
$$Lz_{t+1} = z_t \tag{6}$$

Suppose that previous empirical work suggests the initial values of $\beta = 0.5$, $\kappa = 0.2$, $\rho_u = 0.7$, $\rho_{z1} = 0.7$, and $\rho_{z2} = 0.2$. Inside a scalar substitutable expression, we can specify an initial value by typing {*parameter* = *value*}. We specify these initial values for the parameters in the scalar substitutable expressions in our model specification below.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p    = {beta=.5}*E(F.p) + {kappa=.2}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = (1/{beta})*p + u)
>      (F.u  = {rho_u=.7}*u, state)
>      (F.z  = {rho_z1=.7}*z + {rho_z2=.2}*Lz, state)
>      (F.Lz = z, state noshock)
(setting technique to bfgs)
Iteration 0:   log likelihood = -1117.6457
Iteration 1:   log likelihood = -936.74558  (backed up)
Iteration 2:   log likelihood = -846.69924  (backed up)
Iteration 3:   log likelihood = -835.21274  (backed up)
Iteration 4:   log likelihood = -782.84071  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -772.42535  (not concave)
Iteration 6:   log likelihood =  -754.3824
Iteration 7:   log likelihood = -753.14343
Iteration 8:   log likelihood = -753.07802
Iteration 9:   log likelihood = -753.07788
Iteration 10:  log likelihood = -753.07788
```

```
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.07788
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5154863	.0770497	6.69	0.000	.3644717	.666501
kappa	.1662428	.0468474	3.55	0.000	.0744236	.258062
rho_u	.6979879	.0452072	15.44	0.000	.6093834	.7865923
rho_z1	.6735493	.2666448	2.53	0.012	.150935	1.196164
rho_z2	.270999	.2564998	1.06	0.291	-.2317313	.7737294
sd(e.u)	2.315267	.2992202			1.728806	2.901727
sd(e.z)	.7720681	.1716399			.43566	1.108476

Note that the output includes results for the standard deviations of the shocks, for which there are no scalar substitutable expressions. In example 2, we illustrate how to specify initial values for all the parameters using the `from()` option.

◁

▷ Example 2: Specifying starting values using from()

In example 1, we specified initial values inside the scalar substitutable expressions. You cannot specify initial values for the standard deviations of the shocks this way, because the shocks do not appear in any scalar substitutable expressions. We can specify starting values for any, or all, of the parameters using the `from()` option.

`from()` has several syntaxes, but the vector of starting values is most frequently used in DSGE applications. In this syntax, we specify a row vector containing the initial values $\beta = 0.5$, $\kappa = 0.2$, $\rho_u = 0.7$, $\rho_{z1} = 0.7$, $\rho_{z2} = 0.2$, $\sigma_{e.u} = 2.3$, and $\sigma_{e.z} = 0.7$. We begin by defining and listing the vector of initial values.

```
. matrix ivalues = (.5, .2, .7, .7, .2, 2.3, .7)
. matrix list ivalues
ivalues[1,7]
     c1   c2   c3   c4   c5   c6   c7
r1   .5   .2   .7   .7   .2  2.3   .7
```

Although we could specify and use the column names of the vector to assign elements of the vector to parameters, we use `from`'s suboption `copy` to make the assignment based on order of appearance in the vector. Specifying `from(, copy)` implies that the first element in the vector specifies the initial value for the first model parameter, the second element in the vector specifies the initial value for the second model parameter, and so on.

```
. dsge (p    = {beta}*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = (1/{beta})*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock), from(ivalues, copy)
(setting technique to bfgs)
Iteration 0:   log likelihood = -769.09431
Iteration 1:   log likelihood = -754.09564  (backed up)
Iteration 2:   log likelihood = -753.70352  (backed up)
Iteration 3:   log likelihood = -753.62091  (backed up)
Iteration 4:   log likelihood = -753.53446  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -753.52617  (backed up)
Iteration 6:   log likelihood = -753.09207
Iteration 7:   log likelihood = -753.07789
Iteration 8:   log likelihood = -753.07788
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.07788
```

	Coef.	OIM Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5154871	.0770492	6.69	0.000	.3644734	.6665008
kappa	.1662427	.0468473	3.55	0.000	.0744237	.2580616
rho_u	.6979877	.0452072	15.44	0.000	.6093833	.7865921
rho_z1	.6735477	.2666002	2.53	0.012	.1510209	1.196074
rho_z2	.2710007	.2564571	1.06	0.291	-.2316459	.7736473
sd(e.u)	2.315264	.2992175			1.728808	2.901719
sd(e.z)	.7720676	.1716211			.4356965	1.108439

◁

▷ Example 3: Starting with a more constrained model to obtain convergence

Suppose that we wanted to estimate the parameters of the model in example 1 but that the only good initial value was that theory predicted that β should be about 0.5. Further suppose that we were having convergence problems fitting this model. In this case, one strategy would be to constrain β to 0.5 and to constrain κ to a small, nonzero value. The small value minimizes the interaction between p and y. Making the value nonzero can prevent problems when solving for the state-space form of the model. The remaining parameters determine the autoregressive processes for the states. Given that we suspect that the second-order lag is part of the problem, we also constrain it to a small, nonzero value. We try to estimate the remaining parameters in the output below.

```
. constraint define 1 _b[beta]   = 0.5
. constraint define 2 _b[kappa]  = 0.1
. constraint define 3 _b[rho_z2] = 0.01
. dsge (p    = {beta}*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = (1/{beta})*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock), constraints(1 2 3)
(setting technique to bfgs)
Iteration 0:   log likelihood = -22446.965
Iteration 1:   log likelihood = -2153.9862  (backed up)
Iteration 2:   log likelihood = -1257.6437  (backed up)
Iteration 3:   log likelihood = -1120.3354  (backed up)
Iteration 4:   log likelihood = -1098.8062  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -1092.0434  (not concave)
Iteration 6:   log likelihood =  -865.7853  (not concave)
Iteration 7:   log likelihood =  -781.8036
Iteration 8:   log likelihood = -756.40072
Iteration 9:   log likelihood = -755.63161
Iteration 10:  log likelihood = -755.63025
Iteration 11:  log likelihood = -755.63025
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -755.63025
 ( 1)  [/structural]beta = .5
 ( 2)  [/structural]kappa = .1
 ( 3)  [/structural]rho_z2 = .01
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5	(constrained)				
kappa	.1	(constrained)				
rho_u	.7906612	.0102382	77.23	0.000	.7705947	.8107277
rho_z1	.9440872	.0188166	50.17	0.000	.9072073	.980967
rho_z2	.01	(constrained)				
sd(e.u)	2.391372	.1083319			2.179045	2.603699
sd(e.z)	.697013	.0731968			.5535498	.8404761

The estimator converged. We will want to use these estimates as initial values, so we put them into the matrix `b` and list `b` to confirm it has the desired values.

```
. matrix b = e(b)
. matrix list b
b[1,7]
     /structural:  /structural:  /structural:  /structural:  /structural:
            beta         kappa         rho_u        rho_z1        rho_z2
y1            .5            .1     .79066117     .94408715           .01
                /:            /:
          sd(e.u)       sd(e.z)
y1     2.3913719     .69701298
```

We use the previous estimates as initial values, drop the constraints, and try to estimate all the parameters.

```
. dsge (p    = {beta}*E(F.p) + {kappa}*y)
>      (y    = E(F.y) -(r - E(f.p) - z), unobserved)
>      (r    = (1/{beta})*p + u)
>      (F.u  = {rho_u}*u, state)
>      (F.z  = {rho_z1}*z + {rho_z2}*Lz, state)
>      (F.Lz = z, state noshock), from(b, copy)
(setting technique to bfgs)
Iteration 0:   log likelihood = -755.63025
Iteration 1:   log likelihood = -755.57974  (backed up)
Iteration 2:   log likelihood =  -755.1865  (backed up)
Iteration 3:   log likelihood =  -754.6238  (backed up)
Iteration 4:   log likelihood = -754.55289  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -754.45439  (backed up)
Iteration 6:   log likelihood = -753.14544
Iteration 7:   log likelihood = -753.07913
Iteration 8:   log likelihood = -753.07788
Iteration 9:   log likelihood = -753.07788
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.07788
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5154872	.0770492	6.69	0.000	.3644736	.6665009
kappa	.1662428	.0468472	3.55	0.000	.0744239	.2580617
rho_u	.6979875	.0452071	15.44	0.000	.6093832	.7865918
rho_z1	.6735459	.266585	2.53	0.012	.1510489	1.196043
rho_z2	.2710024	.2564424	1.06	0.291	-.2316156	.7736203
sd(e.u)	2.315263	.2992172			1.728808	2.901718
sd(e.z)	.7720683	.1716153			.4357085	1.108428

The estimator converged.

In larger problems, more steps are sometimes required. In these steps, you use the previous estimates as initial values, but you drop only some of the constraints at each step.

◁

Also see

[DSGE] **intro 5** — Stability conditions

[DSGE] **intro 6** — Identification

Title

intro 8 — Wald tests vary with nonlinear transforms

Description Remarks and examples References Also see

Description

After fitting a DSGE model, we often perform tests of structural parameters, and these tests often place nonlinear restrictions on the parameters. The values and rejection rates of a Wald test for different nonlinear expressions of the same null hypothesis are different. We illustrate this issue, show that likelihood-ratio (LR) tests do not have this problem, and illustrate that you can parameterize your model in terms of invertible transforms of each parameter.

Remarks and examples

Remarks are presented under the following headings:

Wald tests vary with nonlinear transforms

Performing a statistical test of whether a structural parameter in a DSGE has a specific value is one of the most frequent forms of inference after `dsge` estimation. The null hypothesis in one of these tests frequently places nonlinear restrictions on the underlying parameters. Two different nonlinear expressions of the same null hypothesis produce different Wald test statistics in finite samples and have different rejection rates. In other words, the Wald test is not invariant to nonlinear transforms of the null hypothesis. The LR test, on the other hand, is invariant to nonlinear transforms of the null hypothesis.

▷ Example 1: Different values from logically equivalent Wald tests

Equations (1)–(5) specify how the observed control variable inflation p_t, the unobserved control variable output growth y_t, and the observed control variable (interest rate) r_t depend on the states z_t and u_t, given the shocks ϵ_t and ξ_t.

$$p_t = \beta E_t(p_{t+1}) + \kappa y_t \tag{1}$$
$$y_t = E_t(y_{t+1}) - \{r_t - E_t(p_{t+1}) - \rho z_t\} \tag{2}$$
$$r_t = (1/\beta)p_t + u_t \tag{3}$$
$$z_{t+1} = \rho z_t + \epsilon_{t+1} \tag{4}$$
$$u_{t+1} = \delta u_t + \xi_{t+1} \tag{5}$$

We estimate the parameters of this model using the macroeconomic data for the United States in `usmacro2.dta`.

```
. use http://www.stata-press.com/data/r15/usmacro2
(Federal Reserve Economic Data - St. Louis Fed, 2017-01-15)
. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state)
(setting technique to bfgs)
Iteration 0:   log likelihood = -146218.64
Iteration 1:   log likelihood = -5532.4212  (backed up)
Iteration 2:   log likelihood = -1067.4665  (backed up)
Iteration 3:   log likelihood = -938.92415  (backed up)
Iteration 4:   log likelihood = -885.96401  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -880.81744  (not concave)
Iteration 6:   log likelihood = -819.11157  (not concave)
Iteration 7:   log likelihood = -775.50717  (not concave)
Iteration 8:   log likelihood = -766.59529
Iteration 9:   log likelihood = -756.78683  (not concave)
Iteration 10:  log likelihood = -754.40674
Iteration 11:  log likelihood = -753.85297
Iteration 12:  log likelihood = -753.57309
Iteration 13:  log likelihood = -753.57131
Iteration 14:  log likelihood = -753.57131
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5146674	.0783489	6.57	0.000	.3611064	.6682284
kappa	.1659057	.0474074	3.50	0.000	.0729889	.2588224
rhoz	.9545256	.0186424	51.20	0.000	.9179872	.991064
rhou	.7005483	.0452604	15.48	0.000	.6118395	.7892571
sd(e.u)	2.318203	.3047438			1.720916	2.915489
sd(e.z)	.6507118	.1123845			.4304422	.8709814

The interest rate (3) links the nominal interest rate to the inflation rate. The coefficient on inflation is $1/\beta$. We test whether this parameter is 1.5, a common benchmark value in the literature.

```
. testnl 1/_b[beta] = 1.5
  (1)  1/_b[beta] = 1.5
               chi2(1) =        2.24
           Prob > chi2 =        0.1342
```

We do not reject the null hypothesis that $1/\beta$ is 1.5.

If we test the logically equivalent hypothesis that $\beta = 2/3$, the statistic and p-value change.

```
. test _b[beta] =2/3
 ( 1)  [/structural]beta = .6666667
           chi2(  1) =    3.76
         Prob > chi2 =    0.0524
```

The values of these two logically equivalent Wald tests differ because Wald tests are not invariant to nonlinear transformation. This issue is well known in the literature; see Gregory and Veall (1985) and Phillips and Park (1988) for details. In this example, the inference of failing to reject the null hypothesis remains the same when using a 5% significance level, but this is not true in general. Different formulations of Wald tests can lead to different inferences.

◁

LR tests do not vary with nonlinear transforms

LR tests are invariant to nonlinear transforms.

▷ Example 2: LR tests are invariant to nonlinear transforms

We illustrate this feature by performing LR tests that $\beta = 2/3$ and that $1/\beta = 1.5$. The current estimates are those of the unconstrained model. We repeat these results and store them as `unconstrained`.

```
. dsge
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
beta	.5146674	.0783489	6.57	0.000	.3611064	.6682284
kappa	.1659057	.0474074	3.50	0.000	.0729889	.2588224
rhoz	.9545256	.0186424	51.20	0.000	.9179872	.991064
rhou	.7005483	.0452604	15.48	0.000	.6118395	.7892571
sd(e.u)	2.318203	.3047438			1.720916	2.915489
sd(e.z)	.6507118	.1123845			.4304422	.8709814

```
. estimates store unconstrained
```

Now, we estimate the parameters of the constrained model in which $\beta = 2/3$, store the results as `constrained`, and perform an LR test of the null hypothesis that $\beta = 2/3$.

```
. constraint 1 _b[beta] = 2/3

. dsge (p = {beta}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = (1/{beta})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state),
>      constraint(1)
(setting technique to bfgs)
Iteration 0:   log likelihood =  -119695.1
Iteration 1:   log likelihood =  -1425.592  (backed up)
Iteration 2:   log likelihood = -984.57609  (backed up)
Iteration 3:   log likelihood = -948.41524  (backed up)
Iteration 4:   log likelihood = -945.83724  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -945.06881  (backed up)
Iteration 6:   log likelihood = -764.18274  (not concave)
Iteration 7:   log likelihood =  -756.7209
Iteration 8:   log likelihood = -755.12606
Iteration 9:   log likelihood = -755.11016
Iteration 10:  log likelihood = -755.11003
Iteration 11:  log likelihood = -755.11003

DSGE model

Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -755.11003
 ( 1)  [/structural]beta = .6666667

------------------------------------------------------------------------------
             |                 OIM
             |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
/structural  |
        beta |   .6666667  (constrained)
       kappa |   .1076807   .0276892     3.89   0.000     .0534109    .1619506
        rhoz |   .9538519   .0187789    50.79   0.000     .9170458    .9906579
        rhou |   .7214333    .043967    16.41   0.000     .6352595     .807607
-------------+----------------------------------------------------------------
      sd(e.u) |   1.915459   .0867103                      1.74551    2.085408
      sd(e.z) |   .4936815   .0805138                     .3358773    .6514856
------------------------------------------------------------------------------

. estimates store constrained

. lrtest unconstrained constrained

Likelihood-ratio test                                 LR chi2(1)  =      3.08
(Assumption: constrained nested in unconstrained)     Prob > chi2 =    0.0794
```

Note that the value of the LR statistic is 3.08. We now illustrate an LR of the null hypothesis that $1/\beta = 1.5$ produces the same value.

We cannot impose nonlinear restrictions on parameters, so we must begin by reparameterizing the unconstrained model by replacing `{beta}` with `1/{beta}`. To avoid having `{beta}` mean two different things, we write the model in terms of `{gamma}=1/{beta}` and estimate the parameters:

```
. dsge (p = 1/{gamma}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = ({gamma})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state),
>     from(gamma=2 kappa=0.15 rhou=0.75 rhoz=0.95)
(setting technique to bfgs)
Iteration 0:   log likelihood = -1137.8808
Iteration 1:   log likelihood = -1097.9283  (backed up)
Iteration 2:   log likelihood = -1027.9554  (backed up)
Iteration 3:   log likelihood = -801.19555  (backed up)
Iteration 4:   log likelihood = -784.48041  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -763.19407  (not concave)
Iteration 6:   log likelihood = -754.50175  (not concave)
Iteration 7:   log likelihood =  -754.0848
Iteration 8:   log likelihood = -753.57364
Iteration 9:   log likelihood = -753.57131
Iteration 10:  log likelihood = -753.57131
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -753.57131
```

	OIM					
	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
gamma	1.943005	.2957866	6.57	0.000	1.363274	2.522736
kappa	.1659061	.0474073	3.50	0.000	.0729895	.2588226
rhoz	.9545256	.0186424	51.20	0.000	.9179872	.991064
rhou	.7005481	.0452604	15.48	0.000	.6118393	.7892568
sd(e.u)	2.318205	.3047432			1.720919	2.91549
sd(e.z)	.6507123	.1123842			.4304434	.8709812

```
. estimates store unconstrained2
```

The estimates of the parameters other than `gamma` and the value of the log likelihood are nearly the same as those for the `unconstrained` model. The value for `gamma` $= 1.94$ is the same as `1/beta` $= 1/0.514 = 1.95$. By tightening the convergence tolerance, we could make these values exactly the same. These values are nearly the same because this example is an instance of a general property of maximum likelihood estimators. Transforming a parameter by an invertible function does not change the log likelihood or the other parameter estimates. In other words, maximum likelihood estimators are invariant to invertible transformations of the parameters; see Casella and Berger (2002, 319) for details.

Having stored the estimates from the unconstrained model, we now estimate the parameters of the constrained model and store these results in `constrained2`.

```
. constraint 2 _b[gamma] = 1.5

. dsge (p = 1/{gamma}*E(F.p) + {kappa}*y)
>      (y = E(F.y) -(r - E(F.p) - {rhoz}*z), unobserved)
>      (r = ({gamma})*p + u)
>      (F.u = {rhou}*u, state)
>      (F.z = {rhoz}*z, state),
>      constraint(2)
(setting technique to bfgs)
Iteration 0:   log likelihood =  -119695.1
Iteration 1:   log likelihood =  -1425.592  (backed up)
Iteration 2:   log likelihood = -984.57609  (backed up)
Iteration 3:   log likelihood = -948.41524  (backed up)
Iteration 4:   log likelihood = -945.83724  (backed up)
(switching technique to nr)
Iteration 5:   log likelihood = -945.06881  (backed up)
Iteration 6:   log likelihood = -764.18274  (not concave)
Iteration 7:   log likelihood =  -756.7209
Iteration 8:   log likelihood = -755.12606
Iteration 9:   log likelihood = -755.11016
Iteration 10:  log likelihood = -755.11003
Iteration 11:  log likelihood = -755.11003
DSGE model
Sample: 1955q1 - 2015q4                         Number of obs     =        244
Log likelihood = -755.11003
 ( 1)  [/structural]gamma = 1.5
```

	OIM Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
/structural						
gamma	1.5	(constrained)				
kappa	.1076807	.0276892	3.89	0.000	.0534109	.1619506
rhoz	.9538519	.0187789	50.79	0.000	.9170458	.9906579
rhou	.7214333	.043967	16.41	0.000	.6352595	.807607
sd(e.u)	1.915459	.0867103			1.74551	2.085408
sd(e.z)	.4936815	.0805138			.3358773	.6514856

```
. estimates store constrained2
```

The estimates of the parameters other than `gamma` and the value of the log likelihood are the same as those for the `constrained1` model. This is another instance of the invariance of the maximum likelihood estimator to invertible transformations of the parameters.

Having stored the log likelihoods from the constrained and unconstrained model, we now perform an LR of the null hypothesis that $\gamma = 1.5$.

```
. lrtest unconstrained2 constrained2
Likelihood-ratio test                                 LR chi2(1)  =      3.08
(Assumption: constrained2 nested in unconstrained2)   Prob > chi2 =    0.0794
```

The LR test statistic and its p-value are the same as those reported for the test against the null hypothesis that $\beta = 2/3$, which illustrates that LR tests are invariant to nonlinear transforms.

◁

References

Casella, G., and R. L. Berger. 2002. *Statistical Inference*. 2nd ed. Pacific Grove, CA: Duxbury.

Gregory, A. W., and M. R. Veall. 1985. Formulating Wald tests of nonlinear restrictions. *Econometrica* 53: 1465–1468.

Phillips, P. C. B., and J. Y. Park. 1988. On the formulation of Wald tests of nonlinear restrictions. *Econometrica* 56: 1065–1083.

Also see

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[R] **lrtest** — Likelihood-ratio test after estimation

[R] **test** — Test linear hypotheses after estimation

Title

dsge — Linearized dynamic stochastic general equilibrium models

Description

`dsge` fits linearized dynamic stochastic general equilibrium (DSGE) models to multiple time series. DSGE models are systems of equations that are motivated by economic theory. In these systems, expectations of future values of variables can affect the current values. The parameters of these models are often directly interpretable in terms of economic theory.

Menu

Statistics > Multivariate time series > Dynamic stochastic general equilibrium (DSGE) models

Syntax

`dsge` *eqlist* [*if*] [*in*] [, *options*]

options	Description
Main	
`constraints(`*constraints*`)`	apply specified linear constraints
`noidencheck`	do not check for parameter identification
`solve`	return model solution at initial values
SE/Robust	
`vce(`*vcetype*`)`	*vcetype* may be `oim` or `robust`
Reporting	
`level(#)`	set confidence level; default is `level(95)`
`nocnsreport`	do not display constraints
display_options	control columns and column formats and line width
Maximization	
maximize_options	control the maximization process
Advanced	
`nodemean`	do not demean data prior to estimation
`post`	force posting of estimation results in the event of errors caused by lack of identification or stability
`idtolerance(#)`	set tolerance used for identification check; seldom used
`lintolerance(#)`	set tolerance used for linearity check; seldom used
`coeflegend`	display legend instead of statistics

You must `tsset` your data before using `dsge`; see [TS] **tsset**.
`coeflegend` does not appear in the dialog box.
See [U] **20 Estimation and postestimation commands** for more capabilities of estimation commands.

Below we present the full specification of *eqlist*. You may prefer to start with the syntax discussion in [DSGE] **intro 2**.

eqlist is

eq
eq eqlist

eq is

cntrl_eq
state_eq

cntrl_eq contains

(*ocntrl_var* = *termlist*)
(*ucntrl_var* = *termlist*, unobserved)
(*parm_exp* * *ocntrl_var* = *termlist*)
(*parm_exp* * *ucntrl_var* = *termlist*, unobserved)

state_eq contains

(F.*state_var* = , state)
(F.*state_var* = *termlist*, state [noshock])
(*parm_exp* * F.*state_var* = *termlist*, state [noshock])

ocntrl_var is Stata variable to be treated as an observed control in the system
ucntrl_var is name to be treated as an unobserved control in the system
state_var is name to be treated as an unobserved state in the system

If there happens to be a Stata variable with the same name as *ucntrl_var* or *state_var*, the variable is ignored and plays no role in the estimation.

termlist is

term
term + *termlist*
term - *termlist*

term is

rhs_var
parm_exp * *rhs_var*
parm_exp * (*termlist*)

rhs_var is

ocntrl_var
ucntrl_var
state_var
E(F.*ocntrl_var*)
E(F.*ucntrl_var*)

parm_exp is	scalar substitutable expression; *parm_spec* elements are allowed and expected *rhs_var* are not allowed
parm_spec is	{*parm_name*} {*parm_name*=*initial_val*}
parm_name is	name used to identify model parameter
initial_val is	numeric literal; specifies an initial value

Options

Main

`constraints(`*constraints*`)`; see [R] **estimation options**.

`noidencheck` skips the check that the parameters are identified at the initial values. Models that are not structurally identified can still converge, thereby producing meaningless results that only appear to have meaning; thus care should be taken in specifying this option. See [DSGE] **intro 6** for details.

`solve` puts the model into state-space form at the initial parameter values. No standard errors are produced.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are derived from asymptotic theory (`oim`) and that are robust to some kinds of misspecification (`robust`); see [R] ***vce_option***.

Reporting

`level(#)`, `nocnsreport`; see [R] **estimation options**.

display_options: `noci`, `nopvalues`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **estimation options**.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, and `from(`*init_specs*`)`; see [R] **maximize**.

Advanced

`nodemean` prevents `dsge` from removing the mean of the observed control variables prior to estimation.

`post` causes `dsge` to post the parameter vector into `e()`, even in the event of errors that arise from checking stability conditions or identification.

`idtolerance(#)` specifies the tolerance used in the identification diagnostic. The default is `idtolerance(1e-6)`.

`lintolerance(#)` specifies the tolerance used in the linearity diagnostic. The default is `lintolerance(1e-12)`.

The following option is available with dsge but is not shown in the dialog box:

coeflegend; see [R] **estimation options**.

Remarks

For an introduction to what dsge can do and how to use it, see [DSGE] **intro 1**. It is highly recommended that you read the introduction first.

For examples of dsge, see the examples of classic DSGE models in [DSGE] **intro 3a**, [DSGE] **intro 3b**, and [DSGE] **intro 3c**. Additional examples are presented in [DSGE] **intro 4a**–[DSGE] **intro 4g**. See [DSGE] **intro 4** for an overview of these examples.

Stored results

dsge stores the following in e():

Scalars

e(N)	number of observations
e(k)	number of parameters
e(k_aux)	number of auxiliary parameters
e(k_eq)	number of equations in e(b)
e(k_dv)	number of dependent variables
e(k_state)	number of state equations
e(k_control)	number of control equations
e(k_shock)	number of shocks
e(k_observed)	number of observed control equations
e(k_stable)	number of stable eigenvalues
e(ll)	log likelihood
e(tmin)	minimum time in sample
e(tmax)	maximum time in sample
e(rank)	rank of VCE
e(ic)	number of iterations
e(rc)	return code
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	dsge
e(cmdline)	command as typed
e(depvar)	unoperated names of dependent variables
e(state)	unoperated names of state variables
e(control)	unoperated names of control variables
e(observed)	unoperated names of observed control variables
e(title)	title in estimation output
e(tvar)	variable denoting time within groups
e(tmins)	formatted minimum time
e(tmaxs)	formatted maximum time
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. Err.
e(opt)	type of optimization
e(method)	likelihood method
e(idencheck)	passed, failed, or skipped
e(technique)	maximization technique
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(marginsnotok)	predictions disallowed by margins

Matrices

e(b)	parameter vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(eigenvalues)	generalized eigenvalues
e(gradient)	gradient vector
e(shock_coeff)	estimated shock coefficient matrix
e(transition)	estimated state transition matrix
e(policy)	estimated policy matrix
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance

Functions

e(sample)	marks estimation sample

Methods and formulas

A DSGE model is a system of equations in which the values of the control variables $\mathbf{y}_t$ and of the state variables $\mathbf{x}_t$ depend on their values in previous periods and their one-period-ahead rational expectations; see [DSGE] **intro 1** for an introduction to control variables and state variables. The structural form of a DSGE model is

$$\mathbf{A}_0\mathbf{y}_t = \mathbf{A}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{A}_2\mathbf{y}_t + \mathbf{A}_3\mathbf{x}_t \tag{1}$$

$$\mathbf{B}_0\mathbf{x}_{t+1} = \mathbf{B}_1 E_t(\mathbf{y}_{t+1}) + \mathbf{B}_2\mathbf{y}_t + \mathbf{B}_3\mathbf{x}_t + \mathbf{C}\boldsymbol{\epsilon}_{t+1} \tag{2}$$

where the entries in $\mathbf{A}_i$ and $\mathbf{B}_i$ are functions of the parameter vector $\boldsymbol{\theta}$ and $\mathbf{C}$ is a selector matrix of zeros and ones that puts shocks in the state equations that include them. The $\mathbf{A}_0$ and $\mathbf{B}_0$ matrices are diagonal, and $\mathbf{A}_2$ has zeros on its diagonal. See [DSGE] **intro 1** for details about how the structural parameters $\boldsymbol{\theta}$ get mapped into entries in $\mathbf{A}_i$ and $\mathbf{B}_i$ matrices.

dsge estimates the structural parameters $\boldsymbol{\theta}$ by maximum likelihood. The remainder of this entry provides some details about this process.

We can estimate only the parameters of structural models like (1) and (2) that we can solve for the state-space form that expresses the control variables as functions of the state variables and expresses the state variables as first-order autoregressive processes, as in (3) and (4),

$$\mathbf{y}_t = \mathbf{G}(\boldsymbol{\theta})\mathbf{x}_t \tag{3}$$

$$\mathbf{x}_{t+1} = \mathbf{H}(\boldsymbol{\theta})\mathbf{x}_t + \mathbf{M}(\boldsymbol{\theta})\boldsymbol{\epsilon}_{t+1} \tag{4}$$

where $\mathbf{G}$ is known as the policy matrix and $\mathbf{H}$ is known as the state transition matrix. The elements in the policy and state transition matrices are functions of the structural parameters that appear in the matrices $\mathbf{A}$ and $\mathbf{B}$.

dsge implements the solution technique described in Klein (2000), which obtains the state-space form in (3) and (4) from the structural model in (1) and (2).

We can only solve a structural model for its state-space form when i) the structural form has the structure implied by (1) and (2) and when ii) the values of the structural parameters imply that the model neither spirals out of control nor converges to a point, a condition known as saddle-path stability.

Condition (i) seems trivial, but some minor manipulation is frequently required to make it true. Many DSGE models include a problematic term like a shock to a control variable or a lagged state variable. These problematic terms do not fit into the structure required by (1) and (2), but the model is easily rewritten to accommodate these problematic terms; see [DSGE] **intro 4** for details.

Condition (ii), the saddle-path stability condition, depends on the values of the parameters. We can only solve models for parameter values that imply a saddle-path-stable model; see [DSGE] **intro 5** for details.

Assuming that the shocks $\boldsymbol{\epsilon}_t$ are mean-zero, independent, and normally distributed implies that (3) and (4) form a linear state-space model in which the coefficients are nonlinear functions of the structural parameters $\boldsymbol{\theta}$. This structure allows `dsge` to estimate the parameters $\boldsymbol{\theta}$ and the standard deviations of the shocks by maximum likelihood.

The Kalman filter is used to form the log-likelihood function. The Kalman filter is a method for recursively obtaining linear, least-squares forecasts conditional on past information. These forecasts are used to construct the log likelihood, assuming normality and stationarity. See *Methods and formulas* in [TS] **sspace** for details of the Kalman filter.

When the shocks are independent and identically distributed, but not normally distributed, the maximum likelihood estimator is consistent for $\boldsymbol{\theta}$, but `vce(robust)` standard errors must be used. (The process of using a maximum likelihood estimator under weaker distributional assumptions and correcting the standard errors is known as a quasi–maximum likelihood estimator in the literature.)

One final caveat is that only a subset $\mathbf{y}_{1,t}$ of the controls is treated as observed in some models. This caveat is handled by augmenting the state-space systems in (3) and (4) with the observation equation

$$\mathbf{y}_{1,t} = \mathbf{D}\mathbf{y}_t$$

where $\mathbf{D}$ is a selection matrix that picks out the rows in $\mathbf{G}$ corresponding to elements in the control vector that are observed.

Asymptotic standard errors for postestimation objects such as the entries in the state-space matrices and the impulse responses are obtained using the delta method. See Serfling (1980, sec. 3.3) for a discussion of the delta method.

References

Klein, P. 2000. Using the generalized Schur form to solve a multivariate linear rational expectations model. *Journal of Economic Dynamics and Control* 24: 1405–1423.

Serfling, R. J. 1980. *Approximation Theorems of Mathematical Statistics*. New York: Wiley.

Also see

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[DSGE] **intro 2** — Learning the syntax

[TS] **sspace** — State-space models

[TS] **tsset** — Declare data to be time-series data

[TS] **var** — Vector autoregressive models

[U] 20 Estimation and postestimation commands

Title

dsge postestimation — Postestimation tools for dsge

Postestimation commands

The following postestimation commands are of special interest after `dsge`:

Command	Description
`estat policy`	display policy matrix of estimated model
`estat stable`	assess stability of the system
`estat transition`	display transition matrix of estimated model
`irf`	create and analyze IRFs and FEVDs

The following standard postestimation commands are also available:

Command	Description
`estat ic`	Akaike's and Schwarz's Bayesian information criteria (AIC and BIC)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`forecast`	dynamic forecasts and simulations
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of coefficients
`lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	one-step-ahead predictions, prediction errors, and other diagnostic measures
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

predict

Description for predict

`predict` creates new variables containing predictions such as expected values. Predictions are available as static one-step-ahead predictions or as dynamic multistep predictions, and you can control when dynamic predictions begin.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] {*stub*`*`|*newvarlist*} [*if*] [*in*] [`,` *statistic options*]

statistic	Description
Main	
`xb`	linear prediction for observed variables
`states`	linear prediction for latent state variables

options	Description
Options	
`rmse(`*stub*`*`|*newvarlist*`)`	put estimated root mean squared errors of predicted statistics in new variables
`dynamic(`*time_constant*`)`	begin dynamic forecast at specified time
Advanced	
`smethod(`*method*`)`	method for predicting unobserved states

method	Description
`onestep`	predict using past information
`filter`	predict using past and contemporaneous information

Options for predict

Main

`xb`, the default, calculates the linear predictions of the observed variables.

`states` calculates the linear predictions of the latent state variables.

Options

`rmse(`*stub*`*` | *newvarlist*`)` puts the root mean squared errors of the predicted statistics into the specified new variables. The root mean squared errors measure the variances due to the disturbances but do not account for estimation error.

`dynamic(`*time_constant*`)` specifies when `predict` starts producing dynamic forecasts. The specified *time_constant* must be in the scale of the time variable specified in `tsset`, and the *time_constant* must be inside a sample for which observations on the dependent variables are available. For example, `dynamic(tq(2008q4))` causes dynamic predictions to begin in the fourth quarter of 2008, assuming that your time variable is quarterly; see [D] **datetime**. If the model contains exogenous variables, they must be present for the whole predicted sample. `dynamic()` may not be specified with `rstandard`, `residuals`, or `smethod(smooth)`.

Advanced

`smethod(`*method*`)` specifies the method for predicting the unobserved states, `smethod(onestep)` and `smethod(filter)`, and causes different amounts of information on the dependent variables to be used in predicting the states at each time period.

`smethod(onestep)`, the default, causes `predict` to estimate the states at each time period using previous information on the dependent variables. The Kalman filter is performed on previous periods, but only the one-step predictions are made for the current period.

`smethod(filter)` causes `predict` to estimate the states at each time period using previous and contemporaneous data by the Kalman filter. The Kalman filter is performed on previous periods and the current period. `smethod(filter)` may be specified only with `states`.

Remarks and examples

For examples of `estat policy`, see [DSGE] **intro 1**, [DSGE] **intro 3a**, and [DSGE] **intro 3c**.

For examples of `estat transition`, see [DSGE] **intro 1**, [DSGE] **intro 3a**, and [DSGE] **intro 3b**.

For an example of `estat stable`, see [DSGE] **intro 5**.

For examples of `irf` after `dsge`, see [DSGE] **intro 1**, [DSGE] **intro 3b**, and [DSGE] **intro 3c**.

For an example of `forecast` after `dsge`, see [DSGE] **intro 1**.

For examples of `predict` after `dsge`, see [DSGE] **intro 3a**.

Methods and formulas

Estimating the unobserved states is the key to predicting the observed variables.

By default and with the `smethod(onestep)` option, `predict` estimates the state in each period by applying the Kalman filter to all previous periods and only making the one-step prediction to the current period.

With the `smethod(filter)` option, `predict` estimates the states in each period by applying the Kalman filter on all previous periods and the current period. The computational difference between `smethod(onestep)` and `smethod(filter)` is that `smethod(filter)` performs the update step on the current period while `smethod(onestep)` does not. The statistical difference between `smethod(onestep)` and `smethod(filter)` is that `smethod(filter)` uses contemporaneous information on the observed variables while `smethod(onestep)` does not.

The observed control variables are predicted by plugging in the estimated states.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **estat policy** — Display policy matrix

[DSGE] **estat stable** — Check stability of system

[DSGE] **estat transition** — Display state transition matrix

[TS] **forecast** — Econometric model forecasting

[TS] **irf** — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs

[U] **20 Estimation and postestimation commands**

Title

estat policy — Display policy matrix

Description

`estat policy` displays the estimated policy matrix of the state-space form of a DSGE model.

Menu for estat

Statistics > Postestimation

Syntax

`estat policy` [, `level(#)` *display_options*]

Options

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see **[U] 20.8 Specifying the width of confidence intervals**.

display_options: `noci`, `nopvalues`, `cformat(%fmt)`, `pformat(%fmt)`, `sformat(%fmt)`, and `nolstretch`; see [R] **estimation options**.

Remarks and examples

The policy matrix is part of the state-space form of a DSGE model. It specifies the model's control variables as a function of the model's state variables.

For examples, see [DSGE] **intro 1**, [DSGE] **intro 3a**, and [DSGE] **intro 3c**.

Methods and formulas

Entries in the policy matrix $\mathbf{G}$ are functions of the structural parameter vector $\boldsymbol{\theta}$. Standard errors for $\widehat{\mathbf{G}}$ are calculated using the delta method.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[DSGE] **intro 1** — Introduction to DSGEs and dsge

Title

estat stable — Check stability of system

Description

`estat stable` displays functions of the model parameters that indicate whether the model is saddle-path stable at specific parameter values. These results can help you find initial values for which the model is saddle-path stable. Saddle-path stability is required for solving and estimating the parameters of DSGE models.

Menu for estat

Statistics > Postestimation

Syntax

```
estat stable
```

Remarks and examples

DSGE models are dynamic systems that are subject to random shocks. DSGE models that do not spiral out of control or converge to a single point when shocked are said to be "saddle-path stable". We can solve for the state-space form of DSGE models only if they are saddle-path stable, and we can estimate the parameters of models only if we can solve for the state-space form.

The structural parameter values determine whether a DSGE model is saddle-path stable. In the process of solving the structural form for the state-space form, the Klein (2000) solver computes the generalized eigenvalues of a matrix formed from structural parameter values. An eigenvalue is said to be stable when its absolute value is less than 1. The model is saddle-path stable when the number of stable eigenvalues equals the number of states in the model.

`estat stable` displays the generalized eigenvalues implied by the parameter values in `e(b)`, indicates which are stable and which are unstable, and displays a note indicating whether the model is saddle-path stable at these parameter values. `estat stable` can help you find initial values for the maximization routine when the default values imply a model that is not saddle-path stable; see [DSGE] **intro 5** for details.

Methods and formulas

`estat stable` displays the generalized eigenvalues computed by the Klein (2000) solver. Values less than 1 are labeled as stable; values greater than or equal to 1 are labeled as unstable.

Reference

Klein, P. 2000. Using the generalized Schur form to solve a multivariate linear rational expectations model. *Journal of Economic Dynamics and Control* 24: 1405–1423.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[DSGE] **intro 5** — Stability conditions

Title

estat transition — Display state transition matrix

Description

`estat transition` displays the estimated state transition matrix of the state-space form of a DSGE model.

Menu for estat

Statistics > Postestimation

Syntax

```
estat transition [ , level(#) display_options ]
```

Options

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see [U] **20.8 Specifying the width of confidence intervals**.

display_options: `noci`, `nopvalues`, `cformat(%fmt)`, `pformat(%fmt)`, `sformat(%fmt)`, and `nolstretch`; see [R] **estimation options**.

Remarks and examples

The state transition matrix is part of the state-space form of a DSGE model. It specifies the transition matrix of the model's state variables.

For examples, see [DSGE] **intro 1**, [DSGE] **intro 3a**, and [DSGE] **intro 3b**.

Methods and formulas

Entries in the state transition matrix $\mathbf{H}$ are functions of the structural parameter vector $\boldsymbol{\theta}$. Standard errors for entries in $\widehat{\mathbf{H}}$ are calculated using the delta method.

Also see

[DSGE] **dsge** — Linearized dynamic stochastic general equilibrium models

[DSGE] **dsge postestimation** — Postestimation tools for dsge

[DSGE] **intro 1** — Introduction to DSGEs and dsge

Glossary

autoregressive process. An autoregressive process is a time series in which the current value of a variable is a linear function of its own past values and a white-noise error term. A first-order autoregressive process, denoted as an AR(1) process, is $y_t = \rho y_{t-1} + \epsilon_t$. An AR($p$) model contains p lagged values of the dependent variable.

conditional mean. A conditional mean expresses the mean of one variable as a function of some other variables. A regression function is a conditional mean.

control variable. A control variable is an endogenous variable. Control variables can be observed or unobserved.

In a structural DSGE model, the current value of a control variable depends on the current value of other control variables, on the expected future values of any control variable, and on the current values of state variables. The current value of a control variable is found by solving the model for the state-space form.

covariance stationary. A covariance stationary process is a weakly stationary process.

dynamic forecast. A dynamic forecast uses forecasted values wherever lagged values of the endogenous variables appear in the model, allowing one to forecast multiple periods into the future.

dynamic stochastic general equilibrium model. A dynamic stochastic general equilibrium model is a multivariate time-series model that specifies the structural relationship between state variables and control variables and is typically derived from economic theory.

endogenous variable. An endogenous variable is a variable whose values are determined by the equilibrium of a structural model. The values of an endogenous variable are determined inside the system.

equilibrium. The equilibrium values of variables in a model are the values that satisfy all the model's equations simultaneously.

exogenous variable. An exogenous variable is one whose values change independently of the other variables in a structural model. The values of an exogenous variable are determined outside the system. In a time-series model, an exogenous variable is also a predetermined variable.

expected future value. An expected future value is a forecast of the value of a variable in the future based on current information. In DSGE models, expected future values are computed under rational expectations.

Under rational expectations, $E_t(y_{t+1})$ is the condition mean of y_{t+1} conditional on the complete history of all variables in the model and the structure of the model itself.

forward operator. The forward operator F denotes the value of a variable at time $t+1$. Formally, $Fy_t = y_{t+1}$, and $F^2 y_t = Fy_{t+1} = y_{t+2}$.

identified. Identified is a condition required to estimate the parameters of a model. In other words, only identified parameters can be estimated.

In DSGE models, the parameters are identified when there is a unique parameter vector that maximizes the likelihood function. For a discussion of identification, see [DSGE] **intro 6**.

impulse–response function. An impulse–response function (IRF) measures the effect of a shock to an endogenous variable on itself or another endogenous variable. The kth impulse–response function of variable i on variable j measures the effect on variable j in period $t+k$ in response to a one-unit shock to variable i in period t, holding everything else constant.

independent and identically distributed. A series of observations is independent and identically distributed (i.i.d.) if each observation is an independent realization from the same underlying distribution. In some contexts, the definition is relaxed to mean only that the observations are independent and have identical means and variances; see Davidson and MacKinnon (1993, 42).

initial values. Initial values specify the starting place for the iterative maximization algorithm used by DSGE.

Kalman filter. The Kalman filter is a recursive procedure for predicting the state vector in a state-space model.

lag operator. The lag operator L denotes the value of a variable at time $t-1$. Formally, $Ly_t = y_{t-1}$, and $L^2 y_t = Ly_{t-1} = y_{t-2}$.

likelihood-ratio (LR) test. The LR test is a classical testing procedure used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The LR test requires one to determine the maximal value of the log-likelihood function for both the constrained and the full models.

linearized model. A linearized model is an approximation to a model that is nonlinear in the variables and nonlinear in the parameters. The approximation is linear in variables but potentially nonlinear in the parameters.

model solution. A model solution is a function for the endogenous variables in terms of the exogenous variables. A model solution is also known as the reduced form of a model.

In DSGE terminology, a model solution expresses the control variables as a function of the state variables alone and expresses the state variables as a function of their values in the previous period and shocks. The reduced form of a DSGE model is also known as the state-space form of the DSGE model.

model-consistent expectation. A model-consistent expectation is the conditional mean of a variable implied by the model under consideration.

For example, under rational expectations the model-consistent expectation of $E_t(y_{t+1})$ is the mean of y_{t+1} implied by the model, conditional on the realizations of variables dated time t or previously.

nonpredetermined variable. A nonpredetermined variable is a variable whose value at time t is determined by the system of equations in the model. Contrast with predetermined variable.

null hypothesis. In hypothesis testing, the null hypothesis typically represents the conjecture that one is attempting to disprove. Often the null hypothesis is that a parameter is zero or that a statistic is equal across populations.

one-step-ahead forecast. See *static forecast*.

policy matrix. The policy matrix in the reduced form of a DSGE model is the matrix that expresses control variables as a function of state variables.

predetermined variable. A predetermined variable is a variable whose value is fixed at time t, given everything that has occurred previously. More technically, the value of a predetermined variable is fixed, given the realizations of all observed and unobserved variables at times $t-1$, $t-2$, ...

rational expectations. A rational expectation of a variable does not deviate from the mean of that variable in a predictable way. More technically, a rational expectation of a variable is the conditional mean of the variable implied by the model.

realization. The realization of a random variable is the value it takes on when drawn.

reduced form. The reduced form of a model expresses the endogenous variables as functions of the exogenous variables.

The reduced form of a DSGE model expresses the control variables as a function of the state variables alone and expresses the state variables as a function of their values in the previous period and shocks. The reduced form of a DSGE model is a state-space model.

saddle-path stable. A saddle-path stable model is a structural model that can be solved for its state-space form. The existence of a saddle-path stable solution depends on the parameter values of the model. For a discussion of saddle-path stability, see [DSGE] **intro 5**.

shock variable. A shock variable is a random variable whose value is specified as an independently and identically distributed (i.i.d.) random variable. The maximum likelihood estimator is derived under normally distributed shocks but remains consistent under i.i.d. shocks. Robust standard errors must be specified when the errors are i.i.d. but not normally distributed.

state transition matrix. The state transition matrix in the reduced form of a DSGE model is the matrix that expresses how the future values of state variables depend on their current values.

state variable. A state variable is an unobserved exogenous variable.

In DSGE models, a state variable is an unobserved exogenous variable that may depend on its own previous value, the previous values of other state variables, and shocks.

state-space model. A state-space model describes the relationship between an observed time series and an unobservable state vector that represents the "state" of the world. The measurement equation expresses the observed series as a function of the state vector, and the transition equation describes how the unobserved state vector evolves over time. By defining the parameters of the measurement and transition equations appropriately, one can write a wide variety of time-series models in the state-space form.

For DSGE models, the state-space form is the reduced form of the structural model.

The DSGE framework changes the jargon and the structure of state-space models. The measurement equation is the vector of equations for the control variables, and the transition equation is the vector of equations for the state variables. In contrast to the standard state-space model, DSGE models allow a control variable to be unobserved.

static forecast. A static forecast uses actual values wherever lagged values of the endogenous variables appear in the model. As a result, static forecasts perform at least as well as dynamic forecasts, but static forecasts cannot produce forecasts into the future when lags of the endogenous variables appear in the model.

Because actual values will be missing beyond the last historical time period in the dataset, static forecasts can forecast only one period into the future (assuming only first lags appear in the model); thus they are often called one-step-ahead forecasts.

steady-state equilibrium. A steady-state equilibrium is a time-invariant rest point of a dynamic system.

More technically, a steady-state equilibrium is a set of values for the endogenous variables to which the dynamic system will return after an exogenous variable is changed or a random shock occurs. This set of values is time invariant in that it does not depend on the time period in which the change or shock occurs. Multistep dynamic forecasts converge to these values. A steady-state

equilibrium is also known as a long-run equilibrium because it specifies time-invariant values for the endogenous variables to which the dynamic system will return, if left unshocked.

stochastic equation. A stochastic equation, in contrast to an identity, is an equation in a forecast model that includes a random component, most often in the form of an additive error term. Stochastic equations include parameters that must be estimated from historical data.

stochastic trend. A stochastic trend is a nonstationary random process. Unit-root process and random coefficients on time are two common stochastic trends. See [TS] **ucm** for examples and discussions of more commonly applied stochastic trends.

strict stationarity. A process is strictly stationary if the joint distribution of $y_1, \ldots, y_k$ is the same as the joint distribution of $y_{1+\tau}, \ldots, y_{k+\tau}$ for all k and τ. Intuitively, shifting the origin of the series by τ units has no effect on the joint distributions.

structural model. A structural model specifies the theoretical relationship among a set of variables. Structural models contain both endogenous variables and exogenous variables. Parameter estimation and interpretation require that structural models be solved for a reduced form.

trend. The trend specifies the long-run behavior in a time series. The trend can be deterministic or stochastic. Many economic, biological, health, and social time series have long-run tendencies to increase or decrease. Before the 1980s, most time-series analysis specified the long-run tendencies as deterministic functions of time. Since the 1980s, the stochastic trends implied by unit-root processes have become a standard part of the toolkit.

Wald test. A Wald test is a classical testing procedure used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The Wald test requires one to fit the full model but does not require one to fit the constrained model.

weakly stationary. A process is weakly stationary if the mean of the process is finite and independent of t, the unconditional variance of the process is finite and independent of t, and the covariance between periods t and $t - s$ is finite and depends on $t - s$ but not on t or s themselves. Weakly-stationary processes are also known as covariance stationary processes.

white noise. A variable u_t represents a white-noise process if the mean of u_t is zero, the variance of u_t is σ^2, and the covariance between u_t and u_s is zero for all $s \neq t$.

Reference

Davidson, R., and J. G. MacKinnon. 1993. *Estimation and Inference in Econometrics*. New York: Oxford University Press.

Subject and author index

See the combined subject index and the combined author index in the *Glossary and Index*.